



Linux
Professional
Institute

LPIC-1

Phiên bản 5.0
Tiếng Việt

102

Table of Contents

CHỦ ĐỀ 105: VỎ VÀ TẬP LỆNH VỎ	1
105.1 Tùy chỉnh và Sử dụng môi trường vỏ	2
105.1 Bài 1	4
Giới thiệu	4
Các loại Vỏ: Tương tác - Không tương tác và Đăng nhập - Không đăng nhập	5
Bài tập Hướng dẫn	18
Bài tập Mở rộng	20
Tóm tắt	22
Đáp án Bài tập Hướng dẫn	24
Đáp án Bài tập Mở rộng	26
105.1 Bài 2	28
Giới thiệu	28
Biến: Gán và Tham chiếu	28
Biến cục bộ - Biến Vỏ	32
Biến toàn cục - Biến môi trường	34
Bài tập Hướng dẫn	45
Bài tập Mở rộng	48
Tóm tắt	50
Đáp án Bài tập Hướng dẫn	52
Đáp án Bài tập Mở rộng	56
105.1 Bài 3	58
Giới thiệu	58
Tạo Bí danh	58
Tạo Hàm	62
Bài tập Hướng dẫn	73
Bài tập Mở rộng	76
Tóm tắt	77
Đáp án Bài tập Hướng dẫn	79
Đáp án Bài tập Mở rộng	84
105.2 Tùy chỉnh hoặc viết các Tập lệnh đơn giản	85
105.2 Bài 1	87
Giới thiệu	87
Cấu trúc và việc thực thi Tập lệnh	88
Biến	90
Biểu thức Số học	93
Thực thi có điều kiện	94
Đầu ra của Tập lệnh	95
Bài tập Hướng dẫn	98

Bài tập Mở rộng	99
Tóm tắt	100
Đáp án Bài tập Hướng dẫn	101
Đáp án Bài tập Mở rộng	102
105.2 Bài 1	103
Giới thiệu	103
Kiểm tra mở rộng	103
Cấu trúc Vòng lặp	108
Một Ví dụ phức tạp hơn	111
Bài tập Hướng dẫn	115
Bài tập Mở rộng	117
Tóm tắt	118
Đáp án Bài tập Hướng dẫn	119
Đáp án Bài tập Mở rộng	121
CHỦ ĐỀ 106: GIAO DIỆN NGƯỜI DÙNG VÀ MÁY TÍNH ĐỂ BÀN	122
106.1 Cài đặt và định cấu hình X11	123
106.1 Bài 1	124
Giới thiệu	124
Kiến trúc Hệ thống X Window	125
Cấu hình Máy chủ X	128
Wayland	133
Bài tập Hướng dẫn	135
Bài tập Mở rộng	136
Tóm tắt	137
Đáp án Bài tập Hướng dẫn	138
Đáp án Bài tập Mở rộng	139
106.2 Máy tính để bàn đồ họa	140
106.2 Bài 1	141
Giới thiệu	141
Hệ thống X Window	142
Môi trường Máy tính	142
Các Môi trường Máy tính phổ biến	144
Khả năng tương tác trên Máy tính	146
Truy cập phi cục bộ	147
Bài tập Hướng dẫn	150
Bài tập Mở rộng	151
Tóm tắt	152
Đáp án Bài tập Hướng dẫn	153
Đáp án Bài tập Mở rộng	154
106.3 Trợ năng	155

106.3 Bài 1	156
Giới thiệu	156
Cài đặt Trợ năng	156
Hỗ trợ Bàn phím và Chuột	157
Hạn chế về Thị giác	159
Bài tập Hướng dẫn	161
Bài tập Mở rộng	162
Tóm tắt	163
Đáp án Bài tập Hướng dẫn	164
Đáp án Bài tập Mở rộng	165
CHỦ ĐỀ 107: TÁC VỤ QUẢN TRỊ	166
107.1 Quản lý Tài khoản Nhóm và Người dùng cũng như các Tập hệ thống liên quan	167
107.1 Bài 1	169
Giới thiệu	169
Thêm Tài khoản Người dùng	169
Sửa đổi Tài khoản Người dùng	171
Xóa Tài khoản Người dùng	173
Thêm, sửa đổi và xóa Nhóm	173
Thư mục Skeleton	174
Tập /etc/login.defs	174
Lệnh passwd	175
Lệnh chage	176
Bài tập Hướng dẫn	178
Bài tập Mở rộng	179
Tóm tắt	180
Đáp án Bài tập Hướng dẫn	182
Đáp án Bài tập Mở rộng	184
107.1 Bài 2	186
Giới thiệu	186
/etc/passwd	187
/etc/group	188
/etc/shadow	188
/etc/gshadow	189
Lọc Cơ sở dữ liệu Mật khẩu và Nhóm	190
Bài tập Hướng dẫn	191
Bài tập Mở rộng	193
Tóm tắt	194
Đáp án Bài tập Hướng dẫn	195
Đáp án Bài tập Mở rộng	197
107.2 Tự động hóa các Tác vụ Quản trị Hệ thống bằng cách lập lịch trình công việc	199

107.2 Bài 1	201
Giới thiệu	201
Lên lịch công việc với Cron	201
Bảng công việc định kỳ của Người dùng	202
Bảng công việc định kỳ của Hệ thống	203
Thông số thời gian cụ thể	204
Biến Bảng công việc định kỳ	204
Tạo công việc định kỳ của Người dùng	205
Tạo công việc định kỳ cho Hệ thống	206
Định cấu hình Quyền Truy cập vào tác vụ lập lịch trình công việc	207
Một giải pháp thay thế cho Cron	207
Bài tập Hướng dẫn	210
Bài tập Mở rộng	212
Tóm tắt	213
Đáp án Bài tập Hướng dẫn	214
Đáp án Bài tập Mở rộng	216
107.2 Bài 2	218
Giới thiệu	218
Lập lịch trình công việc với at	218
Liệt kê các công việc đã được lập lịch trình với atq	219
Xóa công việc bằng atrm	220
Định cấu hình Quyền truy cập vào tác vụ lập lịch trình công việc	221
Thông số thời gian cụ thể	221
Một giải pháp thay thế cho at	221
Bài tập Hướng dẫn	223
Bài tập Mở rộng	224
Tóm tắt	225
Đáp án Bài tập Hướng dẫn	226
Đáp án Bài tập Mở rộng	227
107.3 Bản địa hoá và Quốc tế hoá	229
107.3 Bài 1	231
Giới thiệu	231
Múi giờ	232
Quy ước Giờ mùa Hè	236
Mã hóa Ngôn ngữ và Ký tự	236
Chuyển đổi Mã hóa	240
Bài tập Hướng dẫn	241
Bài tập Mở rộng	242
Tóm tắt	243
Đáp án Bài tập Hướng dẫn	244

Đáp án Bài tập Mở rộng	245
CHỦ ĐỀ 108: DỊCH VỤ HỆ THỐNG THIẾT YẾU	246
108.1 Duy trì Thời gian Hệ thống	247
108.1 Bài 1	249
Giới thiệu	249
Giờ Địa phương và Giờ Quốc tế	250
Ngày	250
Đồng hồ Phần cứng	252
timedatectl	253
Đặt múi giờ không sử dụng timedatectl	255
Cài đặt ngày giờ không sử dụng timedatectl	255
Bài tập Hướng dẫn	258
Bài tập Mở rộng	260
Tóm tắt	261
Đáp án Bài tập Hướng dẫn	263
Đáp án Bài tập Mở rộng	265
108.1 Bài 2	266
Giới thiệu	266
timedatectl	267
Trình nền NTP	269
Cấu hình NTP	270
pool.ntp.org	271
ntpdate	271
ntpq	271
chrony	272
Bài tập Hướng dẫn	277
Bài tập Mở rộng	279
Tóm tắt	280
Đáp án Bài tập Hướng dẫn	281
Đáp án Bài tập Mở rộng	283
108.2 Ghi nhật ký Hệ thống	284
108.1 Bài 1	286
Giới thiệu	286
Ghi nhật ký Hệ thống	286
Bài tập Hướng dẫn	307
Bài tập Mở rộng	309
Tóm tắt	310
Đáp án Bài tập Hướng dẫn	311
Đáp án Bài tập Mở rộng	314
108.2 Bài 2	315

Giới thiệu	315
Khái niệm cơ bản về systemd	315
Nhật ký hệ thống: systemd- journald	316
Bài tập Hướng dẫn	334
Bài tập Mở rộng	336
Tóm tắt	337
Đáp án Bài tập Hướng dẫn	339
Đáp án Bài tập Mở rộng	342
108.3 Các khái niệm cơ bản về Tác nhân Vận chuyển Thư (MTA)	343
108.1 Bài 1	344
Giới thiệu	344
MTA cục bộ và từ xa	345
MTA của Linux	346
Lệnh mail và Trình duyệt Thư (MUA)	351
Tùy chỉnh phát Thư	352
Bài tập Hướng dẫn	354
Bài tập Mở rộng	355
Tóm tắt	356
Đáp án Bài tập Hướng dẫn	357
Đáp án Bài tập Mở rộng	358
108.4 Quản lý Máy in và tác vụ In	359
108.4 Bài 1	360
Giới thiệu	360
Dịch vụ CUPS	361
Cài đặt một Máy in	365
Quản lý Máy in	367
Gửi Lệnh In	368
Quản lý Lệnh In	371
Xóa máy in	372
Bài tập Hướng dẫn	373
Bài tập Mở rộng	374
Tóm tắt	375
Đáp án Bài tập Hướng dẫn	377
Đáp án Bài tập Mở rộng	378
CHỦ ĐỀ 109: NGUYÊN TẮC CƠ BẢN VỀ MẠNG	380
109.1 Nguyên tắc cơ bản của các Giao thức Internet	381
109.1 Bài 1	382
Giới thiệu	382
IP (Giao thức Internet)	382
Bài tập Hướng dẫn	392

Bài tập Mở rộng	393
Tóm tắt	394
Đáp án Bài tập Hướng dẫn	395
Đáp án Bài tập Mở rộng	396
109.1 Bài 2	397
Giới thiệu	397
Giao thức Điều khiển Truyền nhận (TCP)	399
Giao thức Gói dữ liệu Người dùng (UDP)	399
Giao thức Thông điệp Điều khiển Internet (ICMP)	399
IPv6	399
Bài tập Hướng dẫn	403
Bài tập Mở rộng	404
Tóm tắt	405
Đáp án Bài tập Hướng dẫn	406
Đáp án Bài tập Mở rộng	407
109.2 Cấu hình Mạng liên tục	408
109.2 Bài 1	409
Giới thiệu	409
Giao diện Mạng	409
Tên Giao diện	411
Quản lý Giao diện	412
Tên cục bộ và Tên kết nối từ xa	414
Bài tập Hướng dẫn	418
Bài tập Mở rộng	419
Tóm tắt	420
Đáp án Bài tập Hướng dẫn	421
Đáp án Bài tập Mở rộng	422
109.2 Bài 2	423
Giới thiệu	423
NetworkManager	423
systemd-networkd	427
Bài tập Hướng dẫn	431
Bài tập Mở rộng	432
Tóm tắt	433
Đáp án Bài tập Hướng dẫn	434
Đáp án Bài tập Mở rộng	435
109.3 Xử lý sự cố Mạng cơ bản	436
109.3 Bài 1	438
Giới thiệu	438
Giới thiệu về lệnh ip	439

Nhắc lại về Mặt nạ Mạng và Định tuyến	440
Cấu hình Giao diện	441
Bảng định tuyến	443
Bài tập Hướng dẫn	447
Bài tập Mở rộng	448
Tóm tắt	449
Đáp án Bài tập Hướng dẫn	450
Đáp án Bài tập Mở rộng	452
109.3 Bài 2	454
Giới thiệu	454
Kiểm tra kết nối bằng ping	454
Truy vết Tuyến	455
Tìm MTU bằng tracerpath	458
Tạo Kết nối tùy ý	458
Xem các Kết nối và Trình Nghe hiện tại	460
Bài tập Hướng dẫn	462
Bài tập Mở rộng	463
Tóm tắt	464
Đáp án Bài tập Hướng dẫn	466
Đáp án Bài tập Mở rộng	468
109.4 Định cấu hình DNS phía Máy Khách	470
109.4 Bài 1	471
Giới thiệu	471
Tiến trình phân giải Tên miền	471
Các Hạng DNS	472
Công cụ phân giải Tên miền	475
Bài tập Hướng dẫn	481
Bài tập Mở rộng	482
Tóm tắt	483
Đáp án Bài tập Hướng dẫn	484
Đáp án Bài tập Mở rộng	485
CHỦ ĐỀ 110: BẢO MẬT	487
110.1 Thực hiện nhiệm vụ Quản trị Bảo mật	488
110.1 Bài 1	490
Giới thiệu	490
Kiểm tra tệp bằng bộ SUID và SGID	490
Quản lý Mật khẩu và Tuổi thọ Mật khẩu	493
Khám phá các Cổng mở	496
Giới hạn về Thông tin đăng nhập của Người dùng, Tiến trình và mức sử dụng Bộ nhớ	503
Xử lý những Người dùng đã đăng nhập	506

Cấu hình và cách sử dụng sudo cơ bản	508
Bài tập Hướng dẫn	514
Bài tập Mở rộng	517
Tóm tắt	518
Đáp án Bài tập Hướng dẫn	519
Đáp án Bài tập Mở rộng	524
110.2 Thiết lập Bảo mật Máy chủ	525
110.2 Bài 1	526
Giới thiệu	526
Cải thiện Bảo mật xác thực bằng Mật khẩu ẩn	526
Cách sử dụng Siêu Trình nền để nghe các Kết nối Mạng đến.	528
Kiểm tra các dịch vụ mạng để tìm các trình nền không cần thiết.	533
Trình bao bọc TCP như một loại tường lửa đơn giản.	535
Bài tập Hướng dẫn	536
Bài tập Mở rộng	537
Tóm tắt	538
Đáp án Bài tập Hướng dẫn	540
Đáp án Bài tập Mở rộng	541
110.3 Bảo mật Dữ liệu bằng Mã hóa	542
110.3 Bài 1	544
Giới thiệu	544
Cấu hình và cách sử dụng Máy khách OpenSSH cơ bản	545
Vai trò của Khóa máy chủ của Máy chủ OpenSSH	550
Đường hầm Cổng SSH	552
Bài tập Hướng dẫn	556
Bài tập Mở rộng	558
Tóm tắt	559
Đáp án Bài tập Hướng dẫn	560
Đáp án Bài tập Mở rộng	562
110.3 Bài 2	563
Giới thiệu	563
Thực hiện cấu hình, sử dụng và thu hồi GnuPG cơ bản	563
Sử dụng GPG để mã hóa, giải mã, ký và xác minh tệp	569
Bài tập Hướng dẫn	574
Bài tập Mở rộng	576
Tóm tắt	577
Đáp án Bài tập Hướng dẫn	578
Đáp án Bài tập Mở rộng	580
Ấn bản	581



**Linux
Professional
Institute**

Chủ đề 105: Vỏ và Tập lệnh Vỏ



105.1 Tỳ chỉnh và Sử dụng môi trường vỏ

Tham khảo các mục tiêu LPI

[LPIC-1 version 5.0, Exam 102, Objective 105.1](#)

Khối lượng

4

Các lĩnh vực kiến thức chính

- Đặt các biến môi trường (ví dụ như PATH) khi đăng nhập hoặc khi tạo một vỏ mới.
- Viết các hàm Bash cho các chuỗi lệnh được sử dụng thường xuyên.
- Duy trì các thư mục skeleton cho tài khoản người dùng mới.
- Đặt đường dẫn tìm kiếm lệnh với thư mục thích hợp.

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- `.`
- `source`
- `/etc/bash.bashrc`
- `/etc/profile`
- `env`
- `export`
- `set`
- `unset`
- `~/.bash_profile`
- `~/.bash_login`

- `~/.profile`
- `~/.bashrc`
- `~/.bash_logout`
- `function`
- `alias`



105.1 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	105 Vỏ và Tập lệnh Vỏ
Mục tiêu:	105.1 Tùy chỉnh và Sử dụng môi trường vỏ
Bài:	1 trên 3

Giới thiệu

Vỏ được cho là công cụ mạnh nhất trong hệ thống Linux và có thể được định nghĩa là giao diện giữa người dùng và hạt nhân của hệ điều hành. Vỏ sẽ diễn giải các lệnh được nhập bởi người dùng. Vì vậy, tất cả quản trị viên hệ thống phải có kỹ năng sử dụng vỏ. Chắc hẳn giờ đây chúng ta đều đã biết Bourne Again Shell (*Bash*) chính là vỏ *thực tế* cho phần lớn các bản phân phối Linux.

Một khi được khởi động, điều đầu tiên Bash — hoặc bất kỳ một vỏ nào — sẽ làm là thực thi một loạt các tệp lệnh khởi động. Các tệp lệnh này sẽ tùy chỉnh môi trường của phiên. Sẽ có cả tệp lệnh dành cho toàn hệ thống và tệp lệnh dành riêng cho người dùng. Chúng ta có thể đặt các tùy chọn cá nhân hoặc cài đặt phù hợp nhất với nhu cầu của mình trong các tệp lệnh này dưới dạng biến, bí danh và hàm.

Chuỗi tệp khởi động chính xác sẽ phụ thuộc vào một tham số rất quan trọng là loại vỏ. Hãy cùng xem xét sự đa dạng của những loại vỏ hiện có.

Các loại Vỏ: Tương tác - Không tương tác và Đăng nhập - Không đăng nhập

Để bắt đầu, chúng ta hãy cùng làm rõ các khái niệm về *tương tác* và *đăng nhập* trong ngữ cảnh về vỏ:

Vỏ tương tác / không tương tác

Tên của hai loại vỏ này nói đến sự tương tác diễn ra giữa người dùng và vỏ: người dùng cung cấp đầu vào bằng cách nhập lệnh vào cửa sổ dòng lệnh bằng bàn phím, vỏ sẽ cung cấp đầu ra bằng cách in các thông báo lên trên màn hình.

Vỏ đăng nhập / không đăng nhập

Tên của hai loại vỏ này đề cập đến sự kiện người dùng truy cập vào hệ thống máy tính và cung cấp các thông tin xác thực, chẳng hạn như tên người dùng và mật khẩu.

Cả hai vỏ tương tác và không tương tác đều có thể là vỏ đăng nhập hoặc không đăng nhập và bất kỳ sự kết hợp nào giữa các loại này đều có những cách sử dụng cụ thể.

Vỏ đăng nhập tương tác sẽ được thực thi khi người dùng đăng nhập vào hệ thống. Nó được sử dụng để tùy chỉnh cấu hình của người dùng theo nhu cầu của họ. Một ví dụ điển hình về loại vỏ này là một nhóm người dùng thuộc cùng một bộ phận cần một tập hợp biến cụ thể trong phiên của họ.

Vỏ không đăng nhập tương tác là bất kỳ một vỏ nào khác được người dùng mở sau khi đăng nhập vào hệ thống. Người dùng sử dụng các vỏ này trong các phiên để thực hiện các tác vụ quản trị và bảo trì như thiết lập biến, thời gian, sao chép tệp, viết tệp lệnh, v.v.

Mặt khác, *vỏ không tương tác* sẽ không yêu cầu bất kỳ hình thức tương tác nào từ con người. Do đó, các vỏ này không yêu cầu người dùng nhập dữ liệu đầu vào và đầu ra của chúng - nếu có - trong hầu hết các trường hợp đều sẽ được ghi vào nhật ký.

Các vỏ đăng nhập không tương tác khá là hiếm và không có giá trị sử dụng. Chúng ta gần như không sử dụng đến loại vỏ này và chúng chỉ được nhắc đến để chúng ta hiểu rõ hơn về hành vi của vỏ. Một số ví dụ ngoại lệ bao gồm việc người dùng buộc một tệp lệnh chạy từ một vỏ đăng nhập bằng `/bin/bash --login <some_script>` hoặc chuyển đầu ra tiêu chuẩn (*stdout*) của lệnh vào đầu vào tiêu chuẩn (*stdin*) của một kết nối ssh:

```
<some_command> | ssh <some_user>@<some_server>
```

Vỏ không đăng nhập không tương tác sẽ không có tương tác cũng như không có phiên đăng nhập trên tư cách người dùng, vì thế nên ở đây chúng ta đang nói đến việc sử dụng các tệp lệnh tự động.

Các tập lệnh này chủ yếu được sử dụng để thực hiện các tác vụ quản trị và bảo trì lặp đi lặp lại (chẳng hạn như các tác vụ trong công việc định kỳ). Trong những trường hợp như vậy, `bash` sẽ không đọc bất kỳ tập khởi động nào.

Mở Cửa sổ Dòng lệnh

Trong môi trường máy tính, chúng ta có thể mở một ứng dụng cửa sổ dòng lệnh hoặc chuyển sang một trong các bảng điều khiển hệ thống. Do đó, một vỏ mới có thể là vỏ `pts` khi được mở từ trình mô phỏng cửa sổ dòng lệnh trong GUI hoặc vỏ `tty` khi chạy từ bảng điều khiển hệ thống. Trong trường hợp của vỏ `pts`, chúng ta sẽ không làm việc với cửa sổ dòng lệnh mà với trình mô phỏng cửa sổ dòng lệnh. Là một phần của phiên đồ họa, trình mô phỏng cửa sổ dòng lệnh như *gnome-terminal* hoặc *konsole* rất giàu tính năng và thân thiện với người dùng so với các cửa sổ dòng lệnh có giao diện người dùng thuần văn bản. Trình mô phỏng cửa sổ dòng lệnh có ít tính năng hơn bao gồm *XTerm*, *sakura*, v.v.

Bằng cách sử dụng tổ hợp `Ctrl + Alt + F1-F6`, chúng ta có thể chuyển đến các phiên đăng nhập bảng điều khiển để mở vỏ đăng nhập thuần văn bản tương tác. `Ctrl + Alt + F7` sẽ đưa phiên trở lại màn hình nền.

NOTE

`tty` là viết tắt của teletypewriter (máy đánh chữ) và `pts` là viết tắt của pseudo terminal slave (cửa sổ dòng lệnh giả). Để biết thêm thông tin, hãy sử dụng `man tty` và `man pts`.

Khởi chạy Vỏ bằng `bash`

Sau khi đăng nhập, hãy gõ `bash` vào cửa sổ dòng lệnh để mở một vỏ mới. Về mặt kỹ thuật, vỏ này là một tiến trình con của vỏ hiện tại.

Khi bắt đầu tiến trình con `bash`, chúng ta có thể chỉ định nhiều khoá chuyển khác nhau để xác định loại vỏ mà chúng ta muốn khởi động. Dưới đây là một số tùy chọn gọi `bash` quan trọng:

`bash -l` hoặc `bash --login`

sẽ gọi một vỏ đăng nhập.

`bash -i`

sẽ gọi một vỏ tương tác.

`bash --noprofile`

với vỏ đăng nhập sẽ bỏ qua cả tập khởi động toàn hệ thống (`/etc/profile`) và các tập khởi động cấp người dùng (`~/ .bash_profile`, `~/ .bash_login` và `~/ .profile`).

bash --norc

với các vỏ tương tác sẽ bỏ qua cả tệp khởi động toàn hệ thống (`/etc/bash.bashrc`) và tệp khởi động cấp người dùng (`~/ .bashrc`).

bash --rcfile <file>

với các vỏ tương tác sẽ xem `<file>` như tệp khởi động, bỏ qua `/etc/bash.bashrc` toàn hệ thống và `~/ .bashrc` cấp người dùng.

Chúng ta sẽ cùng thảo luận về các tệp khởi động khác nhau ở bên dưới.

Khởi chạy Vỏ với su và sudo

Thông qua việc sử dụng hai chương trình tương tự nhau này, chúng ta có thể thu được các loại vỏ cụ thể:

su

Thay đổi ID người dùng hoặc trở thành siêu người dùng (`root`). Với lệnh này, chúng ta có thể gọi cả vỏ đăng nhập và không đăng nhập:

- `su - user2`, `su -l user2` hoặc `su --login user2` sẽ bắt đầu một vỏ tương tác đăng nhập với tư cách của `user2`.
- `su user2` sẽ khởi động vỏ tương tác không đăng nhập với tư cách là `user2`.
- `su - root` hoặc `su -` sẽ khởi động vỏ tương tác đăng nhập với tư cách là `root`.
- `su root` hoặc `su` sẽ khởi động một vỏ tương tác không đăng nhập với tư cách là `root`.

sudo

Thực thi các lệnh với tư cách một người dùng khác (bao gồm cả siêu người dùng). Vì lệnh này chủ yếu được sử dụng để lấy quyền gốc tạm thời nên người dùng sử dụng nó phải ở trong tệp `sudoers`. Để thêm người dùng vào `sudoers`, ta cần trở thành `root` và sau đó chạy:

```
root@debian:~# usermod -aG sudo user2
```

Giống như `su`, `sudo` cũng cho phép chúng ta gọi cả vỏ đăng nhập và không đăng nhập:

- `sudo su - user2`, `sudo su -l user2` hoặc `sudo su --login user2` sẽ bắt đầu một vỏ tương tác đăng nhập với tư cách là `user2`.
- `sudo su user2` sẽ khởi động một vỏ tương tác không đăng nhập với tư cách là `user2`.
- `sudo -u user2 -s` sẽ khởi động một vỏ tương tác không đăng nhập với tư cách là `user2`.

- `sudo su - root` hoặc `sudo su -` sẽ khởi động một vỏ tương tác đăng nhập với tư cách là `root`.
- `sudo -i` sẽ khởi động một vỏ tương tác đăng nhập với tư cách là `root`.
- `sudo -i <some_command>` sẽ khởi động một vỏ tương tác đăng nhập với tư cách là `root`, chạy lệnh và quay lại người dùng ban đầu.
- `sudo su root` hoặc `sudo su` sẽ khởi động một vỏ tương tác không đăng nhập với tư cách là `root`.
- `sudo -s` hoặc `sudo -u root -s` sẽ khởi động một vỏ không đăng nhập với tư cách là `root`.

Khi sử dụng `su` hoặc `sudo`, quan trọng nhất là ta phải xem xét trường hợp cụ thể của mình để khởi động một vỏ mới: chúng ta có cần môi trường của người dùng mục tiêu hay không? Nếu có, chúng ta sẽ sử dụng các tùy chọn gọi vỏ đăng nhập; nếu không, chúng ta sẽ sử dụng các tùy chọn gọi vỏ không đăng nhập.

Có những loại Vỏ nào?

Để biết được mình đang làm việc ở loại vỏ nào, chúng ta có thể nhập `echo $0` vào cửa sổ dòng lệnh và nhận được kết quả đầu ra sau:

Đăng nhập tương tác

`-bash` hoặc `-su`

Không đăng nhập tương tác

`bash` hoặc `/bin/bash`

Không đăng nhập không tương tác (tệp lệnh)

`<name_of_script>`

Có bao nhiêu Vỏ?

Để xem có bao nhiêu vỏ `bash` đang chạy trong hệ thống, chúng ta có thể sử dụng lệnh `ps aux | grep bash`:

```
user2@debian:~$ ps aux | grep bash
user2      5270  0.1  0.1 25532  5664 pts/0    Ss   23:03   0:00 bash
user2      5411  0.3  0.1 25608  5268 tty1      S+   23:03   0:00 -bash
user2      5452  0.0  0.0 16760   940 pts/0    S+   23:04   0:00 grep --color=auto bash
```

`user2` tại `debian` đã đăng nhập vào phiên GUI (hoặc hệ thống X Window) và mở `gnome-terminal`,

sau đó họ đã nhấn `Ctrl` + `Alt` + `F1` để chuyển sang một phiên cửa sổ dòng lệnh `tty`. Cuối cùng, họ đã quay lại phiên GUI bằng cách nhấn `Ctrl` + `Alt` + `F7` và gõ lệnh `ps aux | grep bash`. Do đó, đầu ra đã cho thấy một vỏ tương tác không đăng nhập thông qua trình mô phỏng cửa sổ dòng lệnh (`pts/0`) và một vỏ tương tác đăng nhập thông qua cửa sổ dòng lệnh thuần văn bản chuẩn (`tty1`). Cũng hãy lưu ý rằng trường cuối cùng của mỗi dòng (lệnh) là `bash` cho (`pts/0`) và `-bash` cho (`tty1`).

Nơi Vỏ lấy cấu hình: Tập khởi động

Chúng ta đã biết về các loại vỏ có thể tìm thấy trong hệ thống Linux. Đã đến lúc chúng ta biết những tập khởi động nào sẽ được thực thi bởi vỏ nào. Hãy lưu ý rằng các tập lệnh toàn hệ thống hoặc toàn cục sẽ được đặt trong thư mục `/etc/`, trong khi các tập lệnh cục bộ hoặc cấp người dùng sẽ được tìm thấy trong thư mục chính của người dùng (`~`). Ngoài ra, khi cần tìm nhiều hơn một tập, một khi tập được tìm thấy và chạy, các tập khác sẽ bị bỏ qua. Hãy tự mình khám phá và nghiên cứu các tập này bằng trình soạn thảo văn bản yêu thích hoặc bằng cách nhập `less <startup_file>`.

NOTE

Các tập khởi động có thể được chia thành các tập Bash cụ thể (những tập chỉ giới hạn ở cấu hình và lệnh `bash`) và các tập chung (liên quan đến hầu hết các vỏ).

Vỏ Tương tác Đăng nhập

Cấp Toàn cục

`/etc/profile`

Đây là tập `.profile` toàn hệ thống dành cho vỏ Bourne và vỏ tương thích với Bourne (bao gồm `bash`). Thông qua một loạt các câu lệnh `if`, tập này sẽ đặt một số biến tương ứng như `PATH` và `PS1` cũng như tìm nguồn — nếu chúng tồn tại — của cả tập `/etc/bash.bashrc` và các biến trong thư mục `/etc/profile.d`.

`/etc/profile.d/*`

Thư mục này có thể chứa các tập lệnh được thực thi bởi `/etc/profile`.

Cấp cục bộ

`~/.bash_profile`

Tập riêng của Bash này được sử dụng để định cấu hình môi trường người dùng. Nó cũng có thể được sử dụng để lấy nguồn cho cả `~/.bash_login` và `~/.profile`.

`~/.bash_login`

Cũng là tập riêng của Bash. Tập này sẽ chỉ được thực thi nếu không có tập `~/.bash_profile`. Tên của nó gợi ý cho ta biết rằng nó nên được sử dụng để chạy các lệnh cần thiết khi đăng

nhập.

~/ .profile

Tệp này không phải là tệp riêng của Bash và sẽ chỉ được dùng để lấy nguồn nếu cả ~/ .bash_profile và ~/ .bash_login không tồn tại (điều này rất thường xảy ra). Vì vậy, mục đích chính của ~/ .profile là để kiểm tra xem một vỏ Bash có đang được chạy hay không và — nếu đang được chạy — tìm nguồn cho ~/ .bashrc nếu nó tồn tại. Nó thường thiết lập biến PATH để bao gồm thư mục ~/bin riêng tư của người dùng (nếu nó tồn tại).

~/ .bash_logout

Nếu nó tồn tại, tệp riêng của Bash này sẽ thực hiện một số thao tác dọn dẹp khi thoát khỏi vỏ. Điều này sẽ thuận tiện trong những trường hợp như các phiên từ xa.

Khám phá các Tệp cấu hình Vỏ Đăng nhập Tương tác

Hãy cùng xem hoạt động của một số tệp này bằng cách sửa đổi /etc/profile và /home/user2/.profile. Chúng ta sẽ thêm vào mỗi dòng một dòng nhắc nhở về tệp đang được thực thi:

```
root@debian:~# echo 'echo Hello from /etc/profile' >> /etc/profile
root@debian:~# echo 'echo Hello from ~/.profile' >> ~/.profile
```

NOTE

Hai toán tử chuyển hướng >> sẽ nối đầu ra của lệnh vào một tệp hiện có chứ không ghi đè lên nó. Tuy nhiên, nếu tệp không tồn tại, nó sẽ được tạo.

Do đó, thông qua đầu ra của các lệnh echo tương ứng, chúng ta sẽ biết khi nào mỗi tệp này được đọc và thực thi. Để chứng minh cho điều này, hãy cùng xem điều gì sẽ xảy ra khi user2 đăng nhập qua ssh từ một máy khác:

```
user2@debian:~$ ssh user2@192.168.1.6
user2@192.168.1.6's password:
Linux debian 4.9.0-8-amd64 #1 SMP Debian 4.9.130-2 (2018-10-27) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Nov 27 19:57:19 2018 from 192.168.1.10
Hello from /etc/profile
```



```
Hello from /home/user2/.profile
```

Như hai dòng cuối cùng cho biết, nó đã hoạt động. Ngoài ra, hãy lưu ý ba điều:

- Tập toàn cục sẽ được chạy đầu tiên.
- Không có tập `.bash_profile` hoặc `.bash_login` trong thư mục chính của `user2`.
- Dấu ngã (`~`) đã mở rộng đến đường dẫn tuyệt đối của tập (`/home/user2/.profile`).

Vỏ Tương tác không Đăng nhập

Cấp Toàn cục

`/etc/bash.bashrc`

Đây là tập `.bashrc` toàn hệ thống dành cho các vỏ bash tương tác. Thông qua việc thực thi của nó, bash sẽ đảm bảo rằng nó đang được chạy một cách tương tác, kiểm tra kích thước cửa sổ sau mỗi lệnh (cập nhật các giá trị của `LINES` và `COLUMNS` nếu cần) và đặt một số biến.

Cấp Cục bộ

`~/ .bashrc`

Ngoài việc thực hiện các tác vụ tương tự như các tác vụ được mô tả cho `/etc/bash.bashrc` ở cấp độ người dùng (chẳng hạn như kiểm tra kích thước cửa sổ hoặc có đang được chạy tương tác hay không), tập riêng của Bash này thường sẽ thiết lập một số biến lịch sử và tìm nguồn cho `~ / .bash_aliases` nếu nó tồn tại. Ngoài ra, tập này thường được sử dụng để lưu trữ các bí danh và chức năng cụ thể của người dùng.

Tương tự như vậy, cũng cần lưu ý rằng `~/ .bashrc` sẽ được đọc nếu bash phát hiện `<stdin>` của nó là một kết nối mạng (như trường hợp với kết nối *Secure Shell* (SSH) trong ví dụ trên).

Khám phá các Tập cấu hình Vỏ Tương tác không Đăng nhập

Bây giờ, hãy sửa đổi `/etc/bash.bashrc` và `/home/user2/.bashrc`:

```
root@debian:~# echo 'echo Hello from /etc/bash.bashrc' >> /etc/bash.bashrc
root@debian:~# echo 'echo Hello from ~/.bashrc' >> ~/.bashrc
```

Và đây là điều sẽ xảy ra khi `user2` khởi động một vỏ mới:

```
user2@debian:~$ bash
Hello from /etc/bash.bashrc
```

```
Hello from /home/user2/.bashrc
```

Một lần nữa, hai tệp đã được đọc và thực thi.

WARNING

Hãy nhớ rằng do thứ tự chạy các tệp, các tệp cục bộ sẽ được ưu tiên hơn các tệp toàn cục.

Vỏ Đăng nhập không Tương tác

Một vỏ không tương tác với các tùy chọn `-l` hoặc `--login` sẽ buộc phải hoạt động giống như một vỏ đăng nhập và do đó, các tệp khởi động được chạy sẽ giống như các tệp dành cho vỏ tương tác đăng nhập.

Để chứng minh cho điều này, hãy viết một đoạn lệnh đơn giản và làm cho nó có thể thực thi được. Chúng ta sẽ không bao gồm bất kỳ một *shebangs* nào vì ta sẽ gọi *bash executable* (tệp thực thi `bash` - `/bin/bash` với tùy chọn đăng nhập) từ dòng lệnh.

1. Chúng ta sẽ tạo tệp lệnh `test.sh` chứa dòng `echo 'Hello from a script'` để có thể chứng minh rằng tệp lệnh chạy thành công:

```
user2@debian:~$ echo "echo 'Hello from a script'" > test.sh
```

2. Làm cho tệp lệnh có thể thực thi được:

```
user2@debian:~$ chmod +x ./test.sh
```

3. Cuối cùng, chúng ta gọi `bash` với tùy chọn `-l` để chạy tệp lệnh:

```
user2@debian:~$ bash -l ./test.sh
Hello from /etc/profile
Hello from /home/user2/.profile
Hello from a script
```

Nó đã hoạt động! Trước khi chạy tệp lệnh, quá trình đăng nhập đã diễn ra và cả `/etc/profile` và `~/.profile` đều đã được thực thi.

NOTE

Chúng ta sẽ tìm hiểu về *shebangs* và tất cả các khía cạnh khác của tệp lệnh vỏ trong các bài học sau.

Bây giờ, chúng ta đã có đầu ra tiêu chuẩn (*stdout*) của lệnh `echo` vào đầu vào tiêu chuẩn (*stdin*)

của kết nối ssh bằng một đường ống (|):

```
user2@debian:~$ echo "Hello-from-a-noninteractive-login-shell" | ssh user2@192.168.1.6
Pseudo-terminal will not be allocated because stdin is not a terminal.
user2@192.168.1.6's password:
Linux debian 4.9.0-8-amd64 #1 SMP Debian 4.9.130-2 (2018-10-27) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Hello from /etc/profile
Hello from /home/user2/.profile
-bash: line 1: Hello-from-a-noninteractive-login-shell: command not found
```

Một lần nữa, `/etc/profile` và `~/.profile` đã được thực thi. Ngoài ra, dòng đầu tiên và dòng cuối cùng của đầu ra đã nói lên khá nhiều điều về hoạt động của vỏ.

Vỏ không Tương tác không Đăng nhập

Các tệp lệnh sẽ không đọc bất kỳ tệp nào được liệt kê ở trên mà sẽ tìm biến môi trường `BASH_ENV`, mở rộng giá trị của nó nếu cần và sử dụng nó làm tên của tệp khởi động để đọc và thực thi lệnh. Chúng ta sẽ tìm hiểu thêm về *biến môi trường* trong bài học tiếp theo.

Như đã đề cập ở trên, thông thường, `/etc/profile` và `~/.profile` sẽ đảm bảo rằng cả `/etc/bash.bashrc` và `~/.bashrc` đều được thực thi sau khi đăng nhập thành công. Đầu ra của lệnh sau sẽ cho thấy điều này:

```
root@debian:~# su - user2
Hello from /etc/bash.bashrc
Hello from /etc/profile
Hello from /home/user2/.bashrc
Hello from /home/user2/.profile
```

Hãy nhớ rằng các dòng trước đây chúng ta đã thêm vào các tệp lệnh khởi động — và gọi vỏ tương tác đăng nhập ở cấp độ người dùng với `su - user2` — bốn dòng đầu ra có thể được giải thích như sau:

1. `Hello from /etc/bash.bashrc` có nghĩa là `/etc/profile` đã lấy nguồn từ

```
/etc/bash.bashrc.
```

- Hello from /etc/profile có nghĩa là /etc/profile đã được đọc và thực thi đầy đủ.
- Hello from /home/user2/.bashrc có nghĩa là ~/.profile đã lấy nguồn từ ~/.bashrc.
- Hello from /home/user2/.profile có nghĩa là ~/.profile đã được đọc và thực thi đầy đủ.

Hãy lưu ý rằng với `su - <username>` (cũng là `su -l <username>` và `su --login <username>`), chúng ta sẽ đảm bảo được việc gọi vỏ đăng nhập, trong khi `su <username>` chỉ gọi /etc/bash.bashrc và ~/.bashrc.

Lấy nguồn Tập

Trong các phần trước, chúng ta đã thảo luận về việc một số tập lệnh khởi động sẽ bao gồm hoặc thực thi các tập lệnh khác. Cơ chế này được gọi là lấy nguồn và sẽ được giải thích trong phần này.

Lấy nguồn Tập với .

Dấu chấm (.) thường được tìm thấy trong các tập khởi động.

Trong tập .profile của máy chủ Debian, chúng ta có thể tìm thấy — ví dụ — khối sau:

```
# include .bashrc if it exists
if [ -f "$HOME/.bashrc" ]; then
  . "$HOME/.bashrc"
fi
```

Chúng ta đã thấy việc thực thi một tập lệnh có thể dẫn đến việc thực thi một tập lệnh khác như thế nào. Do đó, câu lệnh `if` sẽ đảm bảo rằng tập ` $HOME/.bashrc — nếu nó tồn tại (-f) — sẽ được lấy nguồn (tức là đọc và thực thi) khi đăng nhập:`

```
. "$HOME/.bashrc"
```

NOTE

Như chúng ta sẽ học trong bài học tiếp theo, `$HOME` là một biến môi trường mở rộng đến đường dẫn tuyệt đối của thư mục chính của người dùng.

Hơn nữa, chúng ta có thể sử dụng `.` bất cứ khi nào sửa đổi tập khởi động và muốn thực hiện các thay đổi có hiệu quả mà không cần khởi động lại. Ví dụ: chúng ta có thể:

- thêm bí danh vào ~/.bashrc:

```
user2@debian:~$ echo "alias hi='echo We salute you.'" >> ~/.bashrc
```

WARNING

Khi gửi đầu ra của một lệnh vào một tệp, hãy nhớ đừng nhầm lệnh nối thêm (>>) với lệnh ghi đè (>).

- xuất dòng cuối cùng của `~/.bashrc` để kiểm tra xem mọi thứ có ổn không:

```
user2@debian:~$ tail -n 1 !$
tail -n 1 ~/.bashrc
alias hi='echo We salute you.'
```

NOTE

`!$` mở rộng đến đối số cuối cùng từ lệnh trước đó. Trong trường hợp của chúng ta là `~/.bashrc`.

- lấy nguồn tệp thủ công:

```
user2@debian:~$ . ~/.bashrc
```

- và gọi bí danh để chứng minh là nó có hoạt động:

```
user2@debian:~$ hi
We salute you.
```

NOTE

Tham khảo bài học tiếp theo để tìm hiểu về *bí danh* và *biến*.

Lấy nguồn Tệp với source

Lệnh `source` đồng nghĩa với `.`. Vì vậy, để lấy nguồn `~/.bashrc`, chúng ta cũng có thể làm như sau:

```
user2@debian:~$ source ~/.bashrc
```

Nguồn gốc của các Tệp khởi động Vỏ: SKEL

`SKEL` là một biến với giá trị là đường dẫn tuyệt đối đến thư mục `skel`. Thư mục này phục vụ như một khuôn mẫu cho cấu trúc hệ thống tệp của các thư mục chính của người dùng. Nó bao gồm các tệp sẽ được kế thừa bởi bất kỳ tài khoản người dùng mới nào được tạo (tất nhiên là có bao gồm cả các tệp cấu hình cho vỏ). `SKEL` và các biến liên quan khác được lưu trữ trong `/etc/adduser.conf`

- tệp cấu hình cho adduser:

```
user2@debian:~$ grep SKEL /etc/adduser.conf
# The SKEL variable specifies the directory containing "skeletal" user
SKEL=/etc/skel
# If SKEL_IGNORE_REGEX is set, adduser will ignore files matching this
SKEL_IGNORE_REGEX="dpkg-(old|new|dist|save)"
```

SKEL được đặt thành `/etc/skel`. Do đó, các tệp lệnh khởi động định cấu hình vỏ của chúng ta nằm trong đó:

```
user2@debian:~$ ls -a /etc/skel/
.  ..  .bash_logout  .bashrc  .profile
```

WARNING

Hãy nhớ rằng các tệp bắt đầu bằng `.` sẽ bị ẩn. Vì vậy, chúng ta phải sử dụng `ls -a` để xem chúng khi liệt kê nội dung thư mục.

Bây giờ, hãy tạo một thư mục trong `/etc/skel` để tất cả những người dùng mới có thể lưu trữ tệp lệnh cá nhân của họ trong đó:

1. Với `root`, chúng ta sẽ chuyển vào `/etc/skel`:

```
root@debian:~# cd /etc/skel/
root@debian:/etc/skel#
```

2. Liệt kê nội dung của nó:

```
root@debian:/etc/skel# ls -a
.  ..  .bash_logout  .bashrc  .profile
```

3. Tạo thư mục của mình và kiểm tra xem tất cả có diễn ra như mong đợi:

```
root@debian:/etc/skel# mkdir my_personal_scripts
root@debian:/etc/skel# ls -a
.  ..  .bash_logout  .bashrc  my_personal_scripts  .profile
```

4. Xóa `user2` cùng với thư mục home của nó:

```

root@debian:~# deluser --remove-home user2
Looking for files to backup/remove ...
Removing files ...
Removing user `user2' ...
Warning: group `user2' has no more members.
Done.

```

5. Thêm user2 lần nữa để nó có một thư mục chính mới:

```

root@debian:~# adduser user2
Adding user `user2' ...
Adding new group `user2' (1001) ...
Adding new user `user2' (1001) with group `user2' ...
Creating home directory `/home/user2' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for user2
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] y

```

6. Cuối cùng, đăng nhập với tên user2 và liệt kê tất cả các tệp trong /home/user2 để xem mọi thứ có diễn ra như mong đợi không:

```

root@debian:~# su - user2
user2@debian:~$ pwd
/home/user2
user2@debian:~$ ls -a
.  ..  .bash_history  .bash_logout  .bashrc  my_personal_scripts  .profile

```

Mọi thứ đã diễn ra đúng như chúng ta mong đợi.

Bài tập Hướng dẫn

1. Hãy nghiên cứu cách khởi động các vỏ trong cột “Vỏ khởi động với...” và điền vào các thông tin được yêu cầu:

Vỏ khởi động với...	Tương tác?	Đăng nhập?	Kết quả của echo \$0
sudo ssh user2@machine2			
Ctrl + Alt + F2			
su - user2			
gnome-terminal			
Một người dùng thông thường sử dụng <i>konsole</i> để khởi động một phiên của <i>sakura</i>			
Một tệp lệnh có tên <code>test.sh</code> chứa lệnh <code>echo \$0</code>			

2. Hãy viết lệnh `su` và `sudo` để khởi chạy vỏ được chỉ định:

Vỏ đăng nhập tương tác với tư cách là `user2`

su:

sudo:

Vỏ đăng nhập tương tác với tư cách là `root`

su:

sudo:

Vỏ tương tác không đăng nhập với tư cách là `root`

su:

sudo:

Vỏ tương tác không đăng nhập với tư cách là `user2`

su:

sudo:

3. Tập khởi động nào sẽ được đọc khi vỏ trong cột “Loại vỏ” được khởi động?

Loại vỏ	/etc/profile	/etc/bash.bashrc	~/.profile	~/.bashrc
Vỏ đăng nhập tương tác với tư cách là user2				
Vỏ đăng nhập tương tác với tư cách là root				
Vỏ không đăng nhập tương tác với tư cách là root				
Vỏ không đăng nhập tương tác với tư cách là user2				

Bài tập Mở rộng

1. Trong Bash, chúng ta có thể viết một hàm Hello world! đơn giản bằng cách đưa đoạn mã sau vào một tệp trống:

```
function hello() {  
    echo "Hello world!"  
}
```

- Tiếp theo, chúng ta nên làm gì để đưa hàm này vào vỏ?
- Một khi có sẵn trong vỏ hiện tại, bạn sẽ gọi nó như thế nào?
- Để tự động hóa mọi thứ, bạn sẽ đặt hàm và lệnh gọi của nó vào tệp nào để nó được thực thi khi user2 mở cửa sổ dòng lệnh từ phiên X Window? Đó là loại vỏ gì?
- Bạn sẽ đặt hàm và lệnh gọi của nó vào tệp nào để nó được chạy khi root khởi chạy một vỏ tương tác mới bất kể nó có đăng nhập hay không?

2. Hãy xem đoạn lệnh Hello world! bash cơ bản sau đây:

```
#!/bin/bash  
  
#hello_world: a simple bash script to discuss interaction in scripts.  
  
echo "Xin chào thế giới!"
```

- Giả sử chúng ta làm cho tệp lệnh có thể thực thi được và chạy nó. Đó có phải là một tệp lệnh tương tác không? Tại sao?
- Điều gì làm cho một tệp lệnh có tính tương tác?

3. Hãy tưởng tượng bạn đã thay đổi giá trị của một số biến trong ~/.bashrc và muốn những thay đổi đó có hiệu lực mà không cần khởi động lại. Từ thư mục chính của bạn, làm thế nào để có

thể đạt được điều đó theo hai cách khác nhau?

4. John vừa bắt đầu phiên X Window trên máy chủ Linux. Anh ấy mở một trình mô phỏng cửa sổ dòng lệnh để thực hiện một số nhiệm vụ quản trị, nhưng thật lạ khi phiên này bị treo và anh ấy cần mở một vỏ văn bản.

- Làm thế nào để anh ấy có thể mở được vỏ `ttty` đó?

- Những tệp khởi động nào sẽ được lấy nguồn?

5. Linda là người dùng máy chủ Linux. Cô ấy đã lịch sự yêu cầu quản trị viên có tệp `~/ .bash_login` để cô ấy có thể in ngày và giờ trên màn hình khi đăng nhập. Những người dùng khác rất thích ý tưởng này và đã làm theo. Quản trị viên gặp khó khăn trong việc tạo tệp cho tất cả những người dùng khác trên máy chủ nên anh ấy quyết định thêm một chính sách mới và tạo `~/ .bash_login` cho tất cả những người dùng mới tiềm năng. Làm thế nào để quản trị viên có thể hoàn thành nhiệm vụ này?

Tóm tắt

Trong bài học này, chúng ta đã học về:

- Vỏ sẽ thiết lập môi trường của người dùng trong hệ thống Linux.
- *Bash* là vỏ hàng đầu trên các bản phân phối GNU/Linux.
- Công việc đầu tiên mà vỏ thực hiện là đọc và thực thi một hoặc nhiều tệp khởi động khác nhau.
- Các khái niệm về *tương tác* và *đăng nhập* liên quan đến vỏ.
- Cách khởi chạy các loại vỏ khác nhau bằng `bash`, `su`, `sudo` và `Ctrl` + `Alt` + `F1-F6`.
- Cách kiểm tra loại vỏ bằng `echo $0`.
- Các tệp khởi động cục bộ `~/.bash_profile`, `~/.profile`, `~/.bash_login`, `~/.bash_logout` và `~/.bashrc`.
- Các tệp khởi động toàn cục `/etc/profile`, `/etc/profile.d/*`, `/etc/bash.bashrc`.
- Các tệp cục bộ được ưu tiên hơn các tệp toàn cục.
- Cách chuyển hướng đầu ra của lệnh bằng `>` (ghi đè) và `>>` (nối thêm).
- Ý nghĩa của thư mục `skel`.
- Cách lấy nguồn tệp.

Các lệnh được sử dụng trong bài học này:

bash

Tạo một vỏ mới.

su

Tạo một vỏ mới.

sudo

Tạo một vỏ mới.

usermod

Sửa đổi một tài khoản người dùng.

echo

Hiển thị một dòng văn bản.

ps

Báo cáo ảnh chụp tức thời của các tiến trình hiện tại.

less

Trình đánh số cho các tệp dài.

ssh

Khởi động một kết nối Open SSH (từ xa).

chmod

Thay đổi các bit chế độ của một tệp (ví dụ như làm cho nó có thể thực thi được).

grep

In các dòng phù hợp với một mẫu.

l

Liệt kê nội dung thư mục.

cd

Thay đổi thư mục.

mkdir

Tạo thư mục.

deluser

Xóa người dùng.

Thêm một người dùng mới.

.

Lấy nguồn một tệp.

source

Lấy nguồn một tệp.

tail

Xuất phần cuối cùng của tệp.

Đáp án Bài tập Hướng dẫn

1. Hãy nghiên cứu cách khởi động các vỏ trong cột “Vỏ khởi động với...” và điền vào các thông tin được yêu cầu:

Vỏ khởi động với...	Tương tác?	Đăng nhập?	Kết quả của echo \$0
<code>sudo ssh</code> <code>user2@machine2</code>	Đúng	Đúng	<code>-bash</code>
<code>Ctrl + Alt + F2</code>	Đúng	Đúng	<code>-bash</code>
<code>su - user2</code>	Đúng	Đúng	<code>-bash</code>
<code>gnome-terminal</code>	Đúng	Sai	<code>bash</code>
Một người dùng thông thường sử dụng <code>konsole</code> để khởi động một phiên của <code>sakura</code>	Đúng	Sai	<code>/bin/bash</code>
Một tệp lệnh có tên <code>test.sh</code> chứa lệnh <code>echo \$0</code>	Sai	Sai	<code>./test.sh</code>

2. Hãy viết lệnh `su` và `sudo` để khởi chạy vỏ được chỉ định:

Vỏ đăng nhập tương tác với tư cách là `user2`

`su`

`su - user2`, `su -l user2` hoặc `su --login user2`

`sudo`

`sudo su - user2`, `sudo su -l user2` hoặc `sudo su --login user2`

Vỏ đăng nhập tương tác với tư cách là `root`

`su`

`su - root` hoặc `su -`

`sudo`

`sudo su - root`, `sudo su -` hoặc `sudo -i`

Vỏ tương tác không đăng nhập với tư cách là root**su**`su root` hoặc `su`**sudo**`sudo su root`, `sudo su`, `sudo -s` hoặc `sudo -u root -s`**Vỏ tương tác không đăng nhập với tư cách là user2****su**`su user2`**sudo**`sudo su user2` hoặc `sudo -u user2 -s`

3. Tập khởi động nào sẽ được đọc khi vỏ trong cột “Loại vỏ” được khởi động?

Loại vỏ	<code>/etc/profile</code>	<code>/etc/bash.bashrc</code>	<code>~/.profile</code>	<code>~/.bashrc</code>
Vỏ đăng nhập tương tác với tư cách là user2	Có	Có	Có	Có
Vỏ đăng nhập tương tác với tư cách là root	Có	Có	Không	Không
Vỏ không đăng nhập tương tác với tư cách là root	Không	Có	Không	Không
Vỏ không đăng nhập tương tác với tư cách là user2	Không	Có	Không	Có

Đáp án Bài tập Mở rộng

1. Trong Bash, chúng ta có thể viết một hàm Hello world! đơn giản bằng cách đưa đoạn mã sau vào một tệp trống:

```
function hello() {  
    echo "Hello world!"  
}
```

- Tiếp theo, chúng ta nên làm gì để đưa hàm này vào vỏ?

Để cung cấp hàm cho vỏ hiện tại, chúng ta phải lấy nguồn tệp chứa nó.

- Một khi có sẵn trong vỏ hiện tại, bạn sẽ gọi nó như thế nào?

Chúng ta sẽ gọi nó bằng cách gõ tên của nó vào cửa sổ dòng lệnh.

- Để tự động hóa mọi thứ, bạn sẽ đặt hàm và lệnh gọi của nó vào tệp nào để nó được thực thi khi user2 mở cửa sổ dòng lệnh từ phiên X Window? Đó là loại vỏ gì?

Tệp tốt nhất để đặt nó là `/home/user2/.bashrc`. Vỏ được gọi sẽ là một vỏ tương tác không đăng nhập.

- Bạn sẽ đặt hàm và lệnh gọi của nó vào tệp nào để nó được chạy khi root khởi chạy một vỏ tương tác mới bất kể nó có đăng nhập hay không?

Trong `/etc/bash.bashrc` vì tệp này được thực thi cho tất cả các vỏ tương tác — cho dù có đăng nhập hay không.

2. Hãy xem đoạn lệnh Hello world! bash cơ bản sau đây:

```
#!/bin/bash  
  
#hello_world: a simple bash script to discuss interaction in scripts.  
  
echo "Hello world!"
```

- Giả sử chúng ta làm cho tệp lệnh có thể thực thi được và chạy nó. Đó có phải là một tệp lệnh tương tác không? Tại sao?

Không, vì không có sự tương tác của con người và không có lệnh nào được người dùng nhập

vào.

- Điều gì làm cho một tệp lệnh có tính tương tác?

Việc nó yêu cầu đầu vào từ người dùng.

3. Hãy tưởng tượng bạn đã thay đổi giá trị của một số biến trong `~/ .bashrc` và muốn những thay đổi đó có hiệu lực mà không cần khởi động lại. Từ thư mục chính của bạn, làm thế nào để có thể đạt được điều đó theo hai cách khác nhau?

```
$ source .bashrc
```

hoặc

```
$ . .bashrc
```

4. John vừa bắt đầu phiên X Window trên máy chủ Linux. Anh ấy mở một trình mô phỏng cửa sổ dòng lệnh để thực hiện một số nhiệm vụ quản trị, nhưng thật lạ khi phiên này bị treo và anh ấy cần mở một vỏ văn bản.

- Làm thế nào để anh ấy có thể mở được vỏ `tty` đó?

Anh ấy có thể làm điều đó bằng cách nhấn `Ctrl + Alt + F1-F6` để nhập một trong sáu vỏ `tty`.

- Những tệp khởi động nào sẽ được lấy nguồn?

```
/etc/profile
```

```
/home/john/.profile
```

5. Linda là người dùng máy chủ Linux. Cô ấy đã lịch sự yêu cầu quản trị viên có tệp `~/ .bash_login` để cô ấy có thể in ngày và giờ trên màn hình khi đăng nhập. Những người dùng khác rất thích ý tưởng này và đã làm theo. Quản trị viên gặp khó khăn trong việc tạo tệp cho tất cả những người dùng khác trên máy chủ nên anh ấy quyết định thêm một chính sách mới và tạo `~/ .bash_login` cho tất cả những người dùng mới tiềm năng. Làm thế nào để quản trị viên có thể hoàn thành nhiệm vụ này?

Anh ấy có thể đạt được điều này bằng cách đặt `.bash_login` vào thư mục `/etc/skel`.



105.1 Bài 2

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	105 Vỏ và Tập lệnh Vỏ
Mục tiêu:	105.1 Tùy chỉnh và Sử dụng môi trường vỏ
Bài:	2 trên 3

Giới thiệu

Hãy coi một biến như một chiếc hộp tưởng tượng để tạm thời đặt một mẫu thông tin vào đó. Tương tự như với các tập lệnh khởi tạo của nó, Bash sẽ phân loại các biến thành *vỏ/cục bộ* (những biến chỉ tồn tại trong giới hạn của vỏ mà chúng được tạo) hoặc *môi trường/toàn cục* (những biến được kế thừa bởi vỏ và/hoặc tiến trình con). Trên thực tế, trong bài học trước, chúng ta đã tìm hiểu về vỏ và các tập lệnh cấu hình hoặc khởi tạo của chúng. Thật thuận tiện khi bây giờ chúng ta có thể chỉ ra rằng sức mạnh của các tập khởi động này nằm ở chỗ chúng cho phép chúng ta sử dụng các biến — cũng như các bí danh và hàm — để tạo và tùy chỉnh môi trường vỏ mà chúng ta lựa chọn.

Biến: Gán và Tham chiếu

Một biến có thể được định nghĩa là một cái tên có chứa một giá trị.

Trong Bash, việc đặt giá trị cho tên được gọi là *gán biến* và đó là cách chúng ta tạo hoặc thiết lập biến. Mặt khác, quá trình truy cập giá trị chứa trong tên được gọi là *tham chiếu biến*.

Cú pháp để gán biến là:

```
<variable_name>=<variable_value>
```

Ví dụ:

```
$ distro=zorinos
```

Biến `distro` ngang bằng với `zorinos`, nghĩa là có một phần bộ nhớ giữ giá trị của `zorinos` — với `distro` trở tới nó.

Tuy nhiên, hãy lưu ý rằng khi gán một biến, ở hai bên của dấu bằng không được có bất kỳ một khoảng trắng nào:

```
$ distro =zorinos
-bash: distro: command not found
$ distro= zorinos
-bash: zorinos: command not found
```

Do lỗi của chúng ta, Bash đã đọc `distro` và `zorinos` thành lệnh.

Để tham chiếu một biến (nghĩa là để kiểm tra giá trị của nó), chúng ta sẽ sử dụng lệnh `echo` trước tên biến với dấu `$`:

```
$ echo $distro
zorinos
```

Tên của Biến

Khi chọn tên biến, có một số quy tắc nhất định mà chúng ta phải xem xét.

Tên của biến có thể chứa các chữ cái (a-z,A-Z), số (0-9) và dấu gạch dưới (`_`):

```
$ distro=zorinos
$ echo $distro
zorinos
$ DISTRO=zorinos
$ echo $DISTRO
zorinos
$ distro_1=zorinos
$ echo $distro_1
```

```
zorinos
$ _distro=zorinos
$ echo $_distro
zorinos
```

Tên của biến không được bắt đầu bằng một chữ số vì việc này có thể khiến Bash dễ nhầm lẫn:

```
$ 1distro=zorinos
-bash: 1distro=zorinos: command not found
```

Tên của biến không được chứa dấu cách (thậm chí không được sử dụng cả dấu trích dẫn). Theo quy ước, dấu gạch dưới sẽ được sử dụng thay thế:

```
$ "my distro"=zorinos
-bash: my: command not found
$ my_distro=zorinos
$ echo $my_distro
zorinos
```

Giá trị của Biến

Liên quan đến việc tham chiếu hoặc giá trị của các biến, quan trọng nhất là ta phải xem xét một số quy tắc.

Các biến có thể chứa bất kỳ ký tự chữ và số nào (a-z,A-Z,0-9) cũng như hầu hết những ký tự khác (? , ! , * , . , / , v.v.):

```
$ distro=zorin12.4?
$ echo $distro
zorin12.4?
```

Các giá của trị biến phải được đặt trong dấu trích dẫn nếu chúng chứa các dấu cách đơn:

```
$ distro=zorin 12.4
-bash: 12.4: command not found
$ distro="zorin 12.4"
$ echo $distro
zorin 12.4
$ distro='zorin 12.4'
$ echo $distro
```

```
zorin 12.4
```

Các giá trị của biến cũng phải được đặt trong dấu trích dẫn nếu chúng chứa các ký tự như các ký tự được sử dụng để chuyển hướng (<,>) hoặc ký hiệu dẫn ống (|). Điều duy nhất mà lệnh sau thực hiện là tạo một tệp trống có tên `zorin`:

```
$ distro=>zorin
$ echo $distro

$ ls zorin
zorin
```

Tuy nhiên, khi chúng ta sử dụng dấu trích dẫn thì nó sẽ hoạt động như mong đợi:

```
$ distro=">zorin"
$ echo $distro
>zorin
```

Tuy nhiên, dấu trích dẫn đơn và dấu trích dẫn kép không phải lúc nào cũng có thể hoán đổi cho nhau. Tùy thuộc vào những gì chúng ta đang làm với một biến (gán hoặc tham chiếu), việc sử dụng biến này hay biến kia có ý nghĩa và sẽ mang lại kết quả khác nhau. Khi gán biến, dấu trích dẫn đơn sẽ đọc tất cả các ký tự của giá trị biến *theo nghĩa đen*, trong khi dấu trích dẫn kép sẽ cho phép thay thế biến:

```
$ lizard=uromastyx
$ animal='My $lizard'
$ echo $animal
My $lizard
$ animal="My $lizard"
$ echo $animal
My uromastyx
```

Mặt khác, khi tham chiếu một biến với giá trị có bao gồm một số khoảng trắng đứng đầu (hoặc bổ sung) — đôi khi được kết hợp với dấu hoa thị — chúng ta bắt buộc phải sử dụng dấu trích dẫn kép sau lệnh `echo` để tránh *phân tách trường* và *mở rộng tên đường dẫn*:

```
$ lizard=" genus | uromastyx"
$ echo $lizard
genus | uromastyx
```

```
$ echo "$lizard"
genus | uromastyx
```

Nếu tham chiếu của biến có chứa dấu chấm than đóng thì đây phải là ký tự cuối cùng trong chuỗi (nếu không, Bash sẽ nghĩ rằng chúng ta đang đề cập đến một sự kiện trong quá khứ (history)):

```
$ distro=zorin.?!os
-bash: !os: event not found
$ distro=zorin.?!
$ echo $distro
zorin.?!
```

Mọi dấu gạch chéo ngược đều phải được thoát bằng một dấu gạch chéo ngược khác. Ngoài ra, nếu dấu gạch chéo ngược là ký tự cuối cùng trong chuỗi và chúng ta không thoát nó thì Bash sẽ diễn giải rằng người dùng muốn ngắt dòng và sẽ cung cấp cho chúng ta một dòng mới:

```
$ distro=zorinos\
>
$ distro=zorinos\\
$ echo $distro
zorinos\
```

Trong hai phần tiếp theo, chúng ta sẽ tổng hợp những điểm khác biệt chính giữa biến *cục bộ* và *môi trường*.

Biến cục bộ - Biến Vỏ

Các biến cục bộ hay biến vỏ chỉ tồn tại trong vỏ mà chúng được tạo. Theo quy ước, các biến cục bộ sẽ được viết bằng chữ thường.

Để thực hiện một số thử nghiệm, hãy tạo một biến cục bộ. Như đã giải thích ở trên, chúng ta sẽ chọn một tên biến thích hợp và gán cho nó một giá trị thích hợp. Ví dụ:

```
$ reptile=tortoise
```

Bây giờ, chúng ta sẽ sử dụng lệnh `echo` để tham chiếu biến và kiểm tra xem mọi thứ có diễn ra như mong đợi hay không:

```
$ echo $reptile
```

```
tortoise
```

Trong một số trường hợp nhất định - chẳng hạn như khi viết tệp lệnh - tính bất biến có thể là một tính năng thú vị của các biến. Nếu muốn các biến của mình không thay đổi, chúng ta có thể tạo chúng ở chế độ `readonly` (chỉ đọc):

```
$ readonly reptile=tortoise
```

Hoặc thay đổi sau khi chúng được tạo:

```
$ reptile=tortoise
$ readonly reptile
```

Bây giờ, nếu chúng ta cố gắng thay đổi giá trị của `reptile`, Bash sẽ từ chối:

```
$ reptile=lizard
-bash: distro: readonly variable
```

NOTE

Để liệt kê tất cả các biến chỉ đọc trong phiên hiện tại, hãy nhập `readonly` hoặc `readonly -p` vào cửa sổ dòng lệnh.

Một lệnh hữu ích khi xử lý các biến cục bộ là `set`.

`set` sẽ xuất ra tất cả các biến và hàm của vỏ hiện được gán. Vì nó có thể có rất nhiều dòng (hãy tự mình thử!) nên chúng ta nên sử dụng nó kết hợp với một trình đánh số chẳng hạn như `less`:

```
$ set | less
BASH=/bin/bash
BASHOPTS=checkwinsize:cmdhist:complete_fullquote:expand_aliases:extglob:extquote:force_figno
re:histappend:interactive_comments:login_shell:progcomp:promptvars:sourcepath
BASH_ALIASES=()
BASH_ARGC=()
BASH_ARGV=()
BASH_CMDS=()
BASH_COMPLETION_COMPAT_DIR=/etc/bash_completion.d
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="4" [1]="4" [2]="12" [3]="1" [4]="release" [5]="x86_64-pc-linux-gnu")
BASH_VERSION='4.4.12(1)-release'
```

```
(...)
```

Biến `reptile` có ở đó không?

```
$ set | grep reptile
reptile=tortoise
```

Có!

Tuy nhiên, `reptile` — là một biến cục bộ — sẽ không được kế thừa bởi bất kỳ tiến trình con nào được sinh ra từ vỏ hiện tại:

```
$ bash
$ set | grep reptile
$
```

Và tất nhiên, chúng ta cũng không thể lặp lại giá trị của nó:

```
$ echo $reptile
$
```

NOTE Bằng cách gõ lệnh `bash` vào cửa sổ dòng lệnh, chúng ta sẽ mở một vỏ (con) mới.

Để hủy thiết lập bất kỳ biến nào (cục bộ hoặc toàn cục), chúng ta có thể sử dụng lệnh `unset`:

```
$ echo $reptile
tortoise
$ unset reptile
$ echo $reptile
$
```

NOTE `unset` phải được theo sau bởi tên của biến (không được đặt trước ký hiệu `$`).

Biến toàn cục - Biến môi trường

Các biến toàn cục hoặc biến môi trường tồn tại cho vỏ hiện tại cũng như cho tất cả các tiến trình tiếp theo được sinh ra từ nó. Theo quy ước, các biến môi trường sẽ được viết bằng chữ in hoa:

```
$ echo $SHELL
```



```
/bin/bash
```

Chúng ta có thể truyền đệ quy giá trị của các biến này sang các biến khác và giá trị của biến sau sẽ mở rộng sang giá trị của biến trước:

```
$ my_shell=$SHELL
$ echo $my_shell
/bin/bash
$ your_shell=$my_shell
$ echo $your_shell
/bin/bash
$ our_shell=$your_shell
$ echo $our_shell
/bin/bash
```

Để biến một biến vỏ cục bộ trở thành một biến môi trường, ta phải sử dụng lệnh `export`:

```
$ export reptile
```

Với `export reptile`, chúng ta đã biến biến cục bộ của mình thành biến môi trường để các vỏ con có thể nhận ra và sử dụng nó:

```
$ bash
$ echo $reptile
tortoise
```

Tương tự, `export` có thể được sử dụng để đặt và xuất một biến cùng một lúc:

```
$ export amphibian=frog
```

Bây giờ, chúng ta có thể mở một phiên Bash mới và tham chiếu thành công biến mới:

```
$ bash
$ echo $amphibian
frog
```

NOTE

Với `export -n <VARIABLE-NAME>`, biến sẽ được chuyển trở lại thành biến vỏ cục bộ.

Lệnh `export` cũng sẽ cung cấp cho chúng ta một danh sách tất cả các biến môi trường hiện có khi được gõ riêng (hoặc với tùy chọn `-p`):

```
$ export
declare -x HOME="/home/user2"
declare -x LANG="en_GB.UTF-8"
declare -x LOGNAME="user2"
(...)
declare -x PATH="/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games"
declare -x PWD="/home/user2"
declare -x SHELL="/bin/bash"
declare -x SHLVL="1"
declare -x SSH_CLIENT="192.168.1.10 49330 22"
declare -x SSH_CONNECTION="192.168.1.10 49330 192.168.1.7 22"
declare -x SSH_TTY="/dev/pts/0"
declare -x TERM="xterm-256color"
declare -x USER="user2"
declare -x XDG_RUNTIME_DIR="/run/user/1001"
declare -x XDG_SESSION_ID="8"
declare -x reptile="tortoise"
```

NOTE | Lệnh `declare -x` cũng tương đương với lệnh `export`.

Có hai lệnh nữa có thể được sử dụng để in danh sách tất cả các biến môi trường là `env` và `printenv`:

```
$ env
SSH_CONNECTION=192.168.1.10 48678 192.168.1.7 22
LANG=en_GB.UTF-8
XDG_SESSION_ID=3
USER=user2
PWD=/home/user2
HOME=/home/user2
SSH_CLIENT=192.168.1.10 48678 22
SSH_TTY=/dev/pts/0
MAIL=/var/mail/user2
TERM=xterm-256color
SHELL=/bin/bash
SHLVL=1
LOGNAME=user2
XDG_RUNTIME_DIR=/run/user/1001
PATH=/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
```

```
_=/usr/bin/env
```

Ngoài việc là từ đồng nghĩa với `env`, đôi khi chúng ta có thể sử dụng `printenv` theo cách tương tự khi ta sử dụng lệnh `echo` để kiểm tra giá trị của một biến:

```
$ echo $PWD
/home/user2
$ printenv PWD
/home/user2
```

Tuy nhiên, hãy lưu ý rằng với `printenv`, tên của biến sẽ không có `$` đứng trước.

NOTE

`PWD` lưu trữ đường dẫn của thư mục làm việc hiện tại. Chúng ta sẽ tìm hiểu về điều này và các biến môi trường phổ biến khác sau.

Chạy Chương trình trong một Môi trường đã được sửa đổi

`env` có thể được sử dụng để sửa đổi môi trường vỏ tại thời điểm thực thi chương trình.

Để bắt đầu một phiên vỏ mới với một môi trường trống nhất có thể — xóa sạch hầu hết các biến (cũng như các hàm và bí danh) — chúng ta sẽ sử dụng `env` với tùy chọn `-i`:

```
$ env -i bash
```

Hầu hết các biến môi trường của chúng ta đã biến mất:

```
$ echo $USER
$
```

Và chỉ còn lại một số ít:

```
$ env
LS_COLORS=
PWD=/home/user2
SHLVL=1
_=/usr/bin/printenv
```

Chúng ta cũng có thể sử dụng `env` để đặt một biến cụ thể cho một chương trình cụ thể.

Trong bài học trước, khi thảo luận về *vỏ không đăng nhập không tương tác*, chúng ta đã thấy cách các tập lệnh không đọc bất kỳ tệp khởi động tiêu chuẩn nào mà thay vào đó, chúng sẽ tìm giá trị của biến `BASH_ENV` và sử dụng nó làm tệp khởi động nếu nó tồn tại.

Hãy cùng minh họa quá trình này:

- Chúng ta sẽ tạo tệp khởi động của riêng mình có tên là `.startup_script` với nội dung sau:

```
CROCODILIAN=caiman
```

- Chúng ta sẽ viết một tệp lệnh Bash có tên `test_env.sh` với nội dung sau:

```
#!/bin/bash  
  
echo $CROCODILIAN
```

- Chúng ta sẽ đặt bit thực thi cho tệp lệnh `test_env.sh` của mình:

```
$ chmod +x test_env.sh
```

- Cuối cùng, chúng ta sẽ sử dụng `env` để đặt `BASH_ENV` thành `.startup_script` cho `test_env.sh`:

```
$ env BASH_ENV=/home/user2/.startup_script ./test_env.sh  
caiman
```

Lệnh `env` là lệnh ngầm ngay cả khi chúng ta loại bỏ nó:

```
$ BASH_ENV=/home/user2/.startup_script ./test_env.sh  
caiman
```

NOTE

Nếu không hiểu dòng `#!/bin/bash` hoặc lệnh `chmod +x`, đừng hoảng sợ! Chúng ta sẽ học mọi thứ cần thiết về tệp lệnh vỏ trong các bài học sau. Hiện tại, bạn chỉ cần nhớ rằng để thực thi một tệp lệnh từ trong thư mục riêng của nó, chúng ta sẽ sử dụng `./some_script`.

Biến môi trường chung

Bây giờ là lúc xem lại một số biến môi trường phù hợp nhất được đặt trong tệp cấu hình Bash.

DISPLAY

Liên quan đến máy chủ X, giá trị của biến này thường được tạo thành từ ba phần tử:

- Tên máy chủ (không có nó thì sẽ là `localhost`) nơi máy chủ X đang chạy.
- Dấu hai chấm làm dấu phân cách.
- Một chữ số (thường là `0` và đề cập đến màn hình của máy tính).

```
$ printenv DISPLAY
:0
```

Giá trị trống cho biến này có nghĩa là máy chủ không có Hệ thống X Window. Một số bổ sung — như trong `my.xserver:0:1` — đề cập đến số màn hình (nếu có nhiều hơn một).

HISTCONTROL

Biến này kiểm soát những lệnh nào được lưu vào `HISTFILE` (xem bên dưới). Chúng có ba giá trị có thể có:

ignorespace

Các lệnh bắt đầu bằng dấu cách sẽ không được lưu.

ignoredups

Một lệnh giống với lệnh trước đó sẽ không được lưu.

ignoreboth

Các lệnh thuộc bất kỳ loại nào trong hai loại trước sẽ không được lưu.

```
$ echo $HISTCONTROL
ignoreboth
```

HISTSIZE

Biến này sẽ thiết lập số lượng lệnh được lưu trong bộ nhớ trong suốt phiên vỏ.

```
$ echo $HISTSIZE
1000
```

HISTFILESIZE

Biến này sẽ thiết lập số lượng lệnh được lưu trong HISTFILE cả khi bắt đầu và kết thúc phiên:

```
$ echo $HISTFILESIZE
2000
```

HISTFILE

Tên của tệp lưu trữ tất cả các lệnh khi chúng được gõ. Theo mặc định, tệp này được đặt tại `~/.bash_history`:

```
$ echo $HISTFILE
/home/user2/.bash_history
```

NOTE

Để xem nội dung của HISTFILE, chúng ta chỉ cần gõ `history`. Ngoài ra, chúng ta có thể chỉ định số lượng lệnh mình muốn xem bằng cách truyền một đối số (số lệnh gần đây nhất) cho `history` như trong `history 3`.

HOME

Biến này lưu trữ đường dẫn tuyệt đối của thư mục chính của người dùng hiện tại và nó sẽ được thiết lập khi người dùng đăng nhập.

Đoạn mã này từ `~/.profile` đã tự giải thích chính nó (lấy nguồn từ "`$HOME/.bashrc`" nếu nó tồn tại):

```
# include .bashrc if it exists
if [ -f "$HOME/.bashrc" ]; then
. "$HOME/.bashrc"
fi
```

NOTE

Nếu không hiểu rõ câu lệnh `if` thì cũng đừng lo lắng. Hãy tham khảo các bài học về tập lệnh vỏ.

Hãy nhớ rằng `~` cũng tương đương với `$HOME`:

```
$ echo ~; echo $HOME
/home/carol
/home/carol
```

NOTE

Các lệnh có thể được nối với nhau bằng dấu chấm phẩy (;).

Chúng ta cũng có thể chứng minh điều này bằng câu lệnh `if`:

```
$ if [ ~ == "$HOME" ]; then echo "true"; else echo "false"; fi
true
```

NOTE

Hãy nhớ: Dấu bằng `=` được sử dụng để gán biến, còn `==` được sử dụng để kiểm tra các giá trị tương đương.

HOSTNAME

Biến này lưu trữ tên TCP/IP của máy chủ:

```
$ echo $HOSTNAME
debian
```

HOSTTYPE

Biến này lưu trữ kiến trúc của bộ vi xử lý của máy chủ:

```
$ echo $HOSTTYPE
x86_64
```

LANG

Biến này lưu trữ ngôn ngữ của hệ thống:

```
$ echo $LANG
en_UK.UTF-8
```

LD_LIBRARY_PATH

Biến này bao gồm một tập hợp các thư mục được phân tách bằng dấu hai chấm nơi các thư viện chia sẻ được các chương trình cùng sử dụng:

```
$ echo $LD_LIBRARY_PATH
/usr/local/lib
```

MAIL

Biến này lưu trữ tệp mà Bash tìm kiếm email:

```
$ echo $MAIL
```

```
/var/mail/carol
```

Một giá trị chung khác cho biến này là `/var/spool/mail/$USER`.

MAILCHECK

Biến này lưu trữ một giá trị số cho biết tần suất Bash kiểm tra email mới tính bằng giây:

```
$ echo $MAILCHECK
60
```

PATH

Biến môi trường này lưu trữ danh sách các thư mục nơi Bash sẽ tìm kiếm các tệp thực thi khi được yêu cầu chạy bất kỳ chương trình nào. Trong máy ví dụ của chúng ta, biến này được thiết lập thông qua tệp toàn hệ thống `/etc/profile`:

```
if [ "`id -u`" -eq 0 ]; then
    PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
else
    PATH="/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games"
fi
export PATH
```

Thông qua câu lệnh `if`, danh tính của người dùng sẽ được kiểm tra và — tùy thuộc vào kết quả kiểm tra (root hoặc loại người dùng khác) — chúng ta sẽ nhận được một `PATH` thuộc root hoặc một `PATH` thuộc người dùng khác. Cuối cùng, `PATH` được chọn sẽ được truyền lại bằng `export`.

Hãy quan sát hai điều liên quan đến giá trị của `PATH`:

- Tên thư mục được viết bằng đường dẫn tuyệt đối.
- Dấu hai chấm được dùng làm dấu phân cách.

Nếu muốn đưa thư mục `/usr/local/sbin` vào `PATH` cho người dùng thông thường, chúng ta sẽ sửa đổi dòng sao cho nó trông như dưới đây:

```
(...)
else
    PATH="/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games:/usr/local/sbin"
fi
export PATH
```


Bây giờ, chúng ta có thể thấy giá trị của biến thay đổi như thế nào khi đăng nhập với tư cách người dùng thông thường:

```
# su - carol
$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games:/usr/local/sbin
```

NOTE

Chúng ta cũng có thể thêm `/usr/local/sbin` vào `PATH` của người dùng tại dòng lệnh bằng cách gõ `PATH=/usr/local/sbin:$PATH` hoặc `PATH=$PATH:/usr/local/sbin`. `PATH=/usr/local/sbin:$PATH` sẽ đặt `/usr/local/sbin` làm thư mục đầu tiên được tìm kiếm cho các tệp thực thi. `PATH=$PATH:/usr/local/sbin` sẽ làm cho nó trở thành thư mục cuối cùng.

PS1

Biến này lưu trữ giá trị của dấu nhắc lệnh Bash. Trong đoạn mã sau đây (cũng từ `/etc/profile`), câu lệnh `if` sẽ kiểm tra danh tính của người dùng và cung cấp cho họ một lời nhắc rất riêng biệt tương ứng (`#` cho `root` hoặc `$` cho người dùng thông thường):

```
if [ "`id -u`" -eq 0 ]; then
    PS1='# '
else
    PS1='$ '
fi
```

NOTE

`id` của `root` là `0`. Hãy đăng nhập với tư cách `root` và tự mình kiểm tra nó bằng `id -u`.

Các biến nhắc lệnh khác bao gồm:

PS2

Thường được đặt thành `>` và được sử dụng làm dấu nhắc lệnh tiếp tục cho các lệnh nhiều dòng dài.

PS3

Được sử dụng làm dấu nhắc lệnh cho lệnh `select`.

PS4

Thường được đặt thành `+` và được sử dụng để gỡ lỗi.

SHELL

Biến này lưu trữ đường dẫn tuyệt đối của vỏ hiện tại:

```
$ echo $SHELL  
/bin/bash
```

USER

Biến này lưu trữ tên của người dùng hiện tại:

```
$ echo $USER  
carol
```

Bài tập Hướng dẫn

1. Hãy quan sát việc gán biến trong cột “Lệnh” và cho biết biến kết quả là “Cục bộ” hay “Toàn cục”:

Lệnh	Cục bộ	Toàn cục
debian=mother		
ubuntu=deb-based		
mint=ubuntu-based; export mint		
export suse=rpm-based		
zorin=ubuntu-based		

2. Hãy xem “Lệnh” và “Đầu ra” và giải thích ý nghĩa:

Lệnh	Đầu ra	Ý nghĩa
echo \$HISTCONTROL	ignoreboth	
echo ~	/home/carol	
echo \$DISPLAY	reptilium:0:2	
echo \$MAILCHECK	60	
echo \$HISTFILE	/home/carol/.bash_history	

3. Các biến đang được đặt không chính xác trong cột “Lệnh sai”. Hãy điền thông tin còn thiếu trong cột “Lệnh đúng” và “Tham chiếu biến” để có thể nhận được “Đầu ra dự kiến”:

Lệnh sai	Lệnh đúng	Tham chiếu biến	Đầu ra dự kiến
lizard =chameleon			chameleon
cool lizard=chameleon			chameleon
lizard=cha me leon			cha me leon
lizard=/** chameleon **/			/** chameleon **/

Lệnh sai	Lệnh đúng	Tham chiếu biến	Đầu ra dự kiến
<code>win_path=C:\path\to\dir\</code>			<code>C:\path\to\dir\</code>

4. Hãy đọc mục đích và viết lệnh thích hợp:

Mục đích	Lệnh
Đặt ngôn ngữ của vỏ hiện tại thành UTF-8 tiếng Tây Ban Nha (<code>es_ES.UTF-8</code>).	
In tên của thư mục làm việc hiện tại.	
Tham chiếu biến môi trường lưu trữ thông tin về các kết nối ssh.	
Đặt PATH để <code>/home/carol/scripts</code> làm thư mục cuối cùng khi tìm kiếm các tệp thực thi.	
Đặt giá trị của <code>my_path</code> thành PATH.	
Đặt giá trị của <code>my_path</code> thành giá trị của PATH.	

5. Hãy tạo một biến cục bộ có tên `mammal` và gán cho nó giá trị `gnu`:

6. Bằng cách sử dụng thay thế biến, hãy tạo một biến cục bộ khác có tên `var_sub` với giá trị phù hợp để khi được tham chiếu qua `echo $var_sub`, chúng ta sẽ thu được: `The value of mammal is gnu`:

7. Hãy biến `mammal` thành một biến môi trường:

8. Hãy tìm nó bằng `set` và `grep`:

9. Hãy tìm nó bằng `env` và `grep`:

10. Bằng hai lệnh liên tiếp, hãy tạo một biến môi trường có tên `BIRD` và có giá trị là `penguin`:

11. Bằng một lệnh duy nhất, hãy tạo một biến môi trường có tên `NEW_BIRD` và có giá trị là `yellow-`

eyed penguin:

12. Giả sử bạn là `user2`, hãy tạo một thư mục có tên là `bin` trong thư mục chính của bạn:

13. Hãy nhập lệnh để thêm thư mục `~/bin` vào `PATH` của bạn để nó là thư mục đầu tiên `bash` tìm các tệp nhị phân:

14. Để đảm bảo giá trị của `PATH` không bị thay đổi trong các lần khởi động lại, bạn sẽ đặt đoạn mã nào — dưới dạng câu lệnh `if` — vào `~/.profile`?

Bài tập Mở rộng

1. `let`: hơn cả một lệnh tính biểu thức số học:

- Hãy xem trang hướng dẫn hoặc thực hiện tìm kiếm trên web về `let` và ý nghĩa của nó khi đặt biến và tạo một biến cục bộ mới có tên `my_val` có giá trị là `10` — là kết quả của `5 + 5`:

- Bây giờ, hãy tạo một biến khác có tên `your_val` có giá trị là `5` — là kết quả của việc chia giá trị `my_val` với `2`:

2. Kết quả của một lệnh nằm trong một biến? Tất nhiên là điều này có thể xảy ra. Nó được gọi là *thay thế lệnh*. Hãy tìm hiểu và nghiên cứu hàm có tên `music_info` sau:

```
music_info(){
latest_music=`ls -l1t ~/Music | head -n 6`
echo -e "Your latest 5 music files:\n$latest_music"
}
```

Kết quả của lệnh `ls -l1t ~/Music | head -n 6` sẽ trở thành giá trị của biến `latest_music`. Sau đó, biến `latest_music` sẽ được tham chiếu trong lệnh `echo` (xuất ra tổng số byte bị chiếm bởi thư mục `Music` và năm tệp nhạc mới nhất được lưu trữ trong thư mục `Music` — mỗi tệp một dòng).

Lựa chọn nào sau đây đồng nghĩa hợp lệ với

```
latest_music=`ls -l1t ~/Music | head -n 6`
```

Lựa chọn A:

```
latest_music=$(ls -l1t ~/Music| head -n 6)
```

Lựa chọn B:

```
latest_music="(ls -l1t ~/Music| head -n 6)"
```

Lựa chọn C:

```
latest_music=((ls -l1t ~/Music| head -n 6))
```

Tóm tắt

Trong bài học này, chúng ta đã học về:

- Các biến là một phần rất quan trọng của môi trường vỏ vì chúng được sử dụng bởi chính vỏ cũng như các chương trình khác.
- Cách gán và tham chiếu biến.
- Sự khác biệt giữa các biến *cục bộ* và *toàn cục* (hoặc *môi trường*).
- Cách tạo biến *chỉ đọc*.
- Cách biến biến cục bộ thành biến môi trường bằng lệnh `export`.
- Cách liệt kê tất cả các biến môi trường.
- Cách chạy chương trình trong môi trường đã sửa đổi.
- Cách duy trì các biến với sự trợ giúp của các tập lệnh khởi động.
- Một số biến môi trường phổ biến: `DISPLAY`, `HISTControl`, `HISTSIZE`, `HISTFILESIZE`, `HISTFILE`, `HOME`, `HOSTNAME`, `HOSTTYPE`, `LANG`, `LD_LIBRARY_PATH`, `MAIL`, `MAILCHECK`, `PATH`, `PS1` (và các biến nhắc nhở khác), `SHELL` và `USER`.
- Ý nghĩa của dấu ngã (~).
- Những khái niệm cơ bản về câu lệnh `if`.

Các lệnh được sử dụng trong bài học này:

`echo`

Tham chiếu một biến

`l`

Liệt kê nội dung thư mục.

`readonly`

Làm cho các biến trở nên bất biến. Liệt kê tất cả các biến chỉ đọc trong phiên hiện tại.

`set`

Liệt kê tất cả các biến và hàm trong phiên hiện tại.

`grep`

In các dòng khớp với mẫu.

bash

Khởi chạy một vỏ mới.

unset

Bỏ thiết lập các biến.

export

Biến biến cục bộ thành biến môi trường. Liệt kê các biến môi trường.

env

Liệt kê các biến môi trường. Chạy một chương trình trong một môi trường được sửa đổi.

printenv

Liệt kê các biến môi trường. Tham chiếu một biến

chmod

Thay đổi các bit chế độ của một tệp (ví dụ như làm cho nó có thể thực thi được).

history

Liệt kê các lệnh trước đó.

su

Thay đổi ID người dùng hoặc trở thành siêu người dùng.

id

In ID người dùng.

Đáp án Bài tập Hướng dẫn

1. Hãy quan sát việc gán biến trong cột “Lệnh” và cho biết biến kết quả là “Cục bộ” hay “Toàn cục”:

Lệnh	Cục bộ	Toàn cục
<code>debian=mother</code>	Đúng	Sai
<code>ubuntu=deb-based</code>	Đúng	Sai
<code>mint=ubuntu-based;</code> <code>export mint</code>	Sai	Đúng
<code>export suse=rpm-based</code>	Sai	Đúng
<code>zorin=ubuntu-based</code>	Đúng	Sai

2. Hãy xem “Lệnh” và “Đầu ra” và giải thích ý nghĩa:

Lệnh	Đầu ra	Ý nghĩa
<code>echo \$HISTCONTROL</code>	<code>ignoreboth</code>	Cả hai lệnh đều là lệnh lặp và những lệnh bắt đầu bằng dấu cách sẽ không được lưu trong <code>history</code> .
<code>echo ~</code>	<code>/home/carol</code>	HOME của <code>carol</code> là <code>/home/carol</code> .
<code>echo \$DISPLAY</code>	<code>reptilium:0:2</code>	máy <code>reptilium</code> có máy chủ X đang chạy và chúng ta đang sử dụng màn hình thứ hai của màn hình.
<code>echo \$MAILCHECK</code>	<code>60</code>	Email sẽ được kiểm tra mỗi phút.
<code>echo \$HISTFILE</code>	<code>/home/carol/.bash_history</code>	<code>history</code> sẽ được lưu trong <code>/home/carol/.bash_history</code> .

3. Các biến đang được đặt không chính xác trong cột “Lệnh sai”. Hãy điền thông tin còn thiếu trong cột “Lệnh đúng” và “Tham chiếu biến” để có thể nhận được “Đầu ra dự kiến”:

Lệnh sai	Lệnh đúng	Tham chiếu biến	Đầu ra dự kiến
<code>lizard =chameleon</code>	<code>lizard=chameleon</code>	<code>echo \$lizard</code>	<code>chameleon</code>
<code>cool lizard=chameleon</code>	<code>cool_lizard=chameleon (ví dụ)</code>	<code>echo \$cool_lizard</code>	<code>chameleon</code>
<code>lizard=cha me leon</code>	<code>lizard="cha me leon"</code> hoặc <code>lizard='cha me leon'</code>	<code>echo \$lizard</code>	<code>cha me leon</code>
<code>lizard=/** chameleon **/</code>	<code>lizard="/** chameleon **/"</code> hoặc <code>lizard='/** chameleon **/'</code>	<code>echo "\$lizard"</code>	<code>/** chameleon **/</code>
<code>win_path=C:\path\to\dir\</code>	<code>win_path=C:\\path\\to\\dir\\</code>	<code>echo \$win_path</code>	<code>C:\path\to\dir\</code>

4. Hãy đọc mục đích và viết lệnh thích hợp:

Mục đích	Lệnh
Đặt ngôn ngữ của vỏ hiện tại thành UTF-8 tiếng Tây Ban Nha (<code>es_ES.UTF-8</code>).	<code>LANG=es_ES.UTF-8.</code>
In tên của thư mục làm việc hiện tại.	<code>echo \$PWD</code> hoặc <code>pwd.</code>
Tham chiếu biến môi trường lưu trữ thông tin về các kết nối ssh.	<code>echo \$SSH_CONNECTION.</code>
Đặt PATH để <code>/home/carol/scripts</code> làm thư mục cuối cùng khi tìm kiếm các tệp thực thi.	<code>PATH=\$PATH:/home/carol/scripts.</code>
Đặt giá trị của <code>my_path</code> thành PATH.	<code>my_path=PATH.</code>
Đặt giá trị của <code>my_path</code> thành giá trị của PATH.	<code>my_path=\$PATH.</code>

5. Hãy tạo một biến cục bộ có tên `mammal` và gán cho nó giá trị `gnu`:

```
mammal=gnu
```

6. Bằng cách sử dụng thay thế biến, hãy tạo một biến cục bộ khác có tên `var_sub` với giá trị phù hợp để khi được tham chiếu qua `echo $var_sub`, chúng ta sẽ thu được: `The value of mammal is gnu`:

```
var_sub="The value of mammal is $mammal"
```

7. Hãy biến `mammal` thành một biến môi trường:

```
export mammal
```

8. Hãy tìm nó bằng `set` và `grep`:

```
set | grep mammal
```

9. Hãy tìm nó bằng `set` và `grep`:

```
env | grep mammal
```

10. Bằng hai lệnh liên tiếp, hãy tạo một biến môi trường có tên `BIRD` và có giá trị là `penguin`:

```
BIRD=penguin; export BIRD
```

11. Bằng một lệnh duy nhất, hãy tạo một biến môi trường có tên `NEW_BIRD` và có giá trị là `yellow-eyed penguin`:

```
export NEW_BIRD="yellow-eyed penguin"
```

hoặc

```
export NEW_BIRD='yellow-eyed penguin'
```

12. Giả sử bạn là `user2`, hãy tạo một thư mục có tên là `bin` trong thư mục chính của bạn:

```
mkdir ~/bin
```

hoặc

```
mkdir /home/user2/bin
```

hoặc

```
mkdir $HOME/bin
```

13. Hãy nhập lệnh để thêm thư mục `~/bin` vào `PATH` của bạn để nó là thư mục đầu tiên `bash` tìm các tệp nhị phân:

```
PATH="$HOME/bin:$PATH"
```

`PATH=~/bin:$PATH` hoặc `PATH=/home/user2/bin:$PATH` đều hợp lệ như nhau.

14. Để đảm bảo giá trị của `PATH` không bị thay đổi trong các lần khởi động lại, bạn sẽ đặt đoạn mã nào — dưới dạng câu lệnh `if` — vào `~/ .profile`?

```
if [ -d "$HOME/bin" ] ; then
    PATH="$HOME/bin:$PATH"
fi
```

Đáp án Bài tập Mở rộng

1. `let`: hơn cả một lệnh tính biểu thức số học:

- Hãy xem trang hướng dẫn hoặc thực hiện tìm kiếm trên web về `let` và ý nghĩa của nó khi đặt biến và tạo một biến cục bộ mới có tên `my_val` có giá trị là `10` — là kết quả của `5 + 5`:

```
let "my_val = 5 + 5"
```

hoặc

```
let 'my_val = 5 + 5'
```

- Bây giờ, hãy tạo một biến khác có tên `your_val` có giá trị là `5` — là kết quả của việc chia giá trị `my_val` với `2`:

```
let "your_val = $my_val / 2"
```

hoặc

```
let 'your_val = $my_val / 2'
```

2. Kết quả của một lệnh nằm trong một biến? Tất nhiên là điều này có thể xảy ra. Nó được gọi là *thay thế lệnh*. Hãy tìm hiểu và nghiên cứu hàm có tên `music_info` sau:

```
music_info(){
latest_music=`ls -l1t ~/Music | head -n 6`
echo -e "Your latest 5 music files:\n$latest_music"
}
```

Kết quả của lệnh `ls -l1t ~/Music | head -n 6` sẽ trở thành giá trị của biến `latest_music`. Sau đó, biến `latest_music` sẽ được tham chiếu trong lệnh `echo` (xuất ra tổng số byte được chiếm bởi thư mục `Music` và năm tệp nhạc mới nhất được lưu trữ trong thư mục `Music` — mỗi tệp một dòng).

Lựa chọn nào sau đây đồng nghĩa hợp lệ với

```
latest_music=`ls -l1t ~/Music | head -n 6`
```

Lựa chọn A:

```
latest_music=$(ls -l1t ~/Music| head -n 6)
```



105.1 Bài 3

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	105 Vỏ và Tập lệnh Vỏ
Mục tiêu:	105.1 Tùy chỉnh và Sử dụng môi trường vỏ
Bài:	3 trên 3

Giới thiệu

Sau khi đã đi qua các khái niệm về vỏ, tập lệnh khởi động và các biến trong những bài học trước, chúng ta sẽ kết thúc toàn bộ chủ đề về tùy chỉnh vỏ với hai phần tử rất thú vị của vỏ là *bí danh* và *hàm*. Trên thực tế, biến, bí danh và hàm — cũng như ảnh hưởng của chúng tới nhau — chính là những gì tạo nên môi trường vỏ.

Điểm mạnh chính của hai cơ sở rất linh hoạt và tiết kiệm thời gian này của vỏ chính là sự gói gọn: chúng cung cấp khả năng kết hợp — chỉ bằng một lệnh duy nhất — một loạt các lệnh định kỳ hoặc lặp đi lặp lại.

Tạo Bí danh

Bí danh là một tên khác dùng để thay thế cho (các) lệnh. Theo định nghĩa bí danh, nó có thể chạy như một lệnh thông thường nhưng sẽ thực thi một lệnh khác.

Cú pháp khai báo bí danh khá đơn giản. Bí danh sẽ được khai báo bằng từ khóa `alias`, theo sau là phép gán bí danh. Việc gán bí danh sẽ bao gồm *tên bí danh*, một *dấu bằng* và một hoặc nhiều *lệnh*:


```
alias alias_name=command(s)
```

Ví dụ:

```
$ alias oldshell=sh
```

Bí danh này sẽ bắt đầu một phiên vỏ `sh` ban đầu khi người dùng nhập `oldshell` vào cửa sổ dòng lệnh:

```
$ oldshell
$
```

Sức mạnh của bí danh nằm ở việc chúng giúp ta viết một phiên bản ngắn của các lệnh dài:

```
$ alias ls='ls --color=auto'
```

NOTE

Để biết thông tin về `ls` và màu sắc của nó, hãy nhập `man dir_colors` vào cửa sổ dòng lệnh.

Tương tự như vậy, chúng ta có thể tạo bí danh cho một loạt các lệnh được nối liền nhau — dấu chấm phẩy (;) sẽ được sử dụng làm dấu phân cách. Ví dụ: bí danh có thể cung cấp cho chúng ta thông tin về vị trí của tệp thực thi `git` và phiên bản của nó:

```
$ alias git_info='which git;git --version'
```

Để gọi bí danh, chúng ta sẽ nhập tên của nó vào cửa sổ dòng lệnh:

```
$ git_info
/usr/bin/git
git version 2.7.4
```

Lệnh `alias` sẽ tạo một danh sách tất cả các bí danh có sẵn trong hệ thống:

```
$ alias
alias git-info='which git;git --version'
alias ls='ls --color=auto'
alias oldshell='sh'
```

Lệnh `unalias` sẽ loại bỏ bí danh. Ví dụ: chúng ta có thể loại bỏ bí danh `git-info` và xem nó biến mất khỏi danh sách như thế nào:

```
$ unalias git-info
$ alias
alias ls='ls --color=auto'
alias oldshell='sh'
```

Như đã thấy với `alias hi='echo We salute you.'` trong bài học trước, chúng ta phải đặt các lệnh trong dấu trích dẫn (đơn hoặc kép) khi — do có đối số hoặc tham số — chúng có chứa dấu cách:

```
$ alias greet='echo Hello world!'
$ greet
Hello world!
```

Các lệnh có dấu cách cũng bao gồm các lệnh có tùy chọn:

```
$ alias ll='ls -al'
```

Bây giờ, `ll` sẽ liệt kê tất cả các tệp — bao gồm cả các tệp bị ẩn (`a`) — ở định dạng chi tiết (`l`).

Chúng ta có thể tham chiếu các biến trong bí danh:

```
$ reptile=uromastyx
$ alias greet='echo Hello $reptile!'
$ greet
Hello uromastyx!
```

Biến cũng có thể được gán trong bí danh:

```
$ alias greet='reptile=tortoise; echo Hello $reptile!'
$ greet
Hello tortoise!
```

Chúng ta có thể thoát bí danh bằng `\`:

```
$ alias where?='echo $PWD'
```

```
$ where?
/home/user2
$ \where?
-bash: where?: command not found
```

Thoát bí danh rất hữu ích khi bí danh có cùng tên với một lệnh thông thường. Trong trường hợp này, bí danh sẽ được ưu tiên hơn lệnh ban đầu. Tuy nhiên, lệnh này vẫn có thể truy cập được bằng cách thoát bí danh.

Tương tự như vậy, chúng ta có thể đặt một bí danh bên trong một bí danh khác:

```
$ where?
/home/user2
$ alias my_home=where?
$ my_home
/home/user2
```

Ngoài ra, chúng ta cũng có thể đặt một hàm bên trong bí danh như chúng ta sẽ thấy ở dưới đây.

Tính năng mở rộng và đánh giá của Dấu trích dẫn khi sử dụng với Bí danh

Khi sử dụng dấu trích dẫn với các biến môi trường, dấu trích dẫn đơn sẽ làm cho việc mở rộng trở nên động:

```
$ alias where?='echo $PWD'
$ where?
/home/user2
$ cd Music
$ where?
/home/user2/Music
```

Tuy nhiên, với dấu trích dẫn kép, việc mở rộng sẽ được thực hiện tĩnh:

```
$ alias where?="echo $PWD"
$ where?
/home/user2
$ cd Music
$ where?
/home/user2
```

Tính bền của Bí danh: Tập lệnh khởi động

Cũng giống như các biến, để bí danh có được tính bền bỉ, ta phải đặt chúng vào các tập lệnh khởi tạo được lấy nguồn khi khởi động. Như chúng ta đã biết, tập phù hợp để người dùng đặt bí danh cá nhân là `~/.bashrc`. Ta cũng có thể sẽ tìm thấy một số bí danh có sẵn ở đó (hầu hết chúng đều đã được vô hiệu hoá dưới dạng chú thích và sẵn sàng được sử dụng bằng cách xóa `#` ở đầu):

```
$ grep alias .bashrc
# enable color support of ls and also add handy aliases
  alias ls='ls --color=auto'
  #alias dir='dir --color='
  #alias vdir='vdir --color='
  #alias grep='grep --color='
  #alias fgrep='fgrep --color='
  #alias egrep='egrep --color='
# some more ls aliases
#ll='ls -al'
#alias la='ls -A'
#alias l='ls -CF'
# ~/.bash_aliases, instead of adding them here directly.
if [ -f ~/.bash_aliases ]; then
  . ~/.bash_aliases
```

Như có thể thấy trong ba dòng cuối cùng, hệ thống cho phép người dùng có một tệp dành riêng cho bí danh — `~/.bash_aliases` — và lấy nguồn cho nó từ `.bashrc` mỗi khi hệ thống khởi động. Vì vậy, chúng ta có thể lựa chọn phương án này để tạo và điền vào tệp đó:

```
#####
# .bash_aliases:
# a file to be populated by the user's personal aliases (and sourced by ~/.bashrc).
#####
alias git_info='which git;git --version'
alias greet='echo Hello world!'
alias ll='ls -al'
alias where?='echo $PWD'
```

Tạo Hàm

So với bí danh, các hàm có tính lập trình và linh hoạt hơn, đặc biệt là trong việc khai thác toàn bộ tiềm năng của các biến tích hợp sẵn đặc biệt của *Bash* và các tham số vị trí. Chúng cũng làm việc rất tốt với các cấu trúc kiểm soát luồng như vòng lặp hoặc cấu trúc điều kiện. Chúng ta có thể coi

hàm như một lệnh bao gồm logic thông qua các khối hoặc tập hợp các lệnh khác.

Hai cú pháp để tạo Hàm

Có hai cú pháp hợp lệ để xác định hàm.

Sử dụng từ khóa `function`

Một mặt, chúng ta có thể sử dụng từ khóa `function`, theo sau là tên hàm và các lệnh nằm trong ngoặc nhọn:

```
function function_name {
command #1
command #2
command #3
.
.
.
command #n
}
```

Sử dụng `()`

Mặt khác, chúng ta cũng có thể bỏ từ khóa `function` và thay vào đó là sử dụng hai dấu ngoặc đơn ngay sau tên của hàm:

```
function_name() {
command #1
command #2
command #3
.
.
.
command #n
}
```

Việc đặt các hàm trong tệp hoặc tệp lệnh là điều rất bình thường. Tuy nhiên, chúng cũng có thể được ghi trực tiếp vào dấu nhắc lệnh vỏ với mỗi lệnh trên một dòng khác nhau — lưu ý rằng PS2(>) biểu thị một dòng mới sau một dấu ngắt dòng:

```
$ greet() {
> greeting="Hello world!"
```

```
> echo $greeting
> }
```

Dù trong trường hợp nào — và bất kể cú pháp chúng ta chọn là gì —, nếu chúng ta quyết định bỏ qua dấu ngắt dòng và viết một hàm chỉ trong một dòng thì các lệnh phải được phân tách bằng dấu chấm phẩy (hãy lưu ý cả dấu chấm phẩy ở sau lệnh cuối cùng):

```
$ greet() { greeting="Hello world!"; echo $greeting; }
```

`bash` đã không hề có bất cứ phần nào khi chúng ta nhấn Enter, vì vậy nên hàm của chúng ta đã sẵn sàng được gọi. Để gọi một hàm, chúng ta phải nhập tên của nó vào cửa sổ dòng lệnh:

```
$ greet
Hello world!
```

Cũng giống như các biến và bí danh, nếu muốn các hàm tồn tại bền bỉ trong suốt quá trình khởi động lại hệ thống, chúng ta phải đặt chúng vào các tệp lệnh khởi tạo vỏ như `/etc/bash.bashrc` (toàn cục) hoặc `~/.bashrc` (cục bộ).

WARNING

Sau khi thêm bí danh hoặc hàm vào bất kỳ tệp tập lệnh khởi động nào, chúng ta cũng nên cấp nguồn cho các tệp đó bằng `.` hoặc `source` để các thay đổi có hiệu lực (trong trường hợp bạn không muốn đăng xuất và đăng nhập lại hoặc khởi động lại hệ thống).

Các Biến tích hợp sẵn đặc biệt của Bash

Vỏ Bourne Again đi kèm với một tập hợp các biến đặc biệt rất hữu ích cho các hàm và tệp lệnh. Chúng đặc biệt vì chỉ có thể được tham chiếu chứ không được chỉ định. Dưới đây là danh sách các biến quan trọng nhất:

\$?

Tham chiếu của biến này sẽ mở rộng đến kết quả của lần chạy lệnh cuối cùng. Giá trị `0` có nghĩa là đã thành công:

```
$ ps aux | grep bash
user2      420  0.0  0.4 21156  5012 pts/0    Ss   17:10   0:00 -bash
user2      640  0.0  0.0 12784   936 pts/0    S+   18:04   0:00 grep bash
$ echo $?
0
```

Giá trị khác 0 có nghĩa là đã gặp lỗi:

```
user1@debian:~$ ps aux |rep bash
-bash: rep: command not found
user1@debian:~$ echo $?
127
```

\$\$

Biến mở rộng sang vỏ PID (ID tiến trình):

```
$ ps aux | grep bash
user2      420  0.0  0.4 21156 5012 pts/0    Ss   17:10   0:00 -bash
user2      640  0.0  0.0 12784  936 pts/0    S+   18:04   0:00 grep bash
$ echo $$
420
```

\$!

Biến mở rộng đến PID của tác vụ chạy ngầm cuối cùng:

```
$ ps aux | grep bash &
[1] 663
$ user2      420  0.0  0.4 21156 5012 pts/0    Ss+  17:10   0:00 -bash
user2      663  0.0  0.0 12784  972 pts/0    S    18:08   0:00 grep bash
^C
[1]+  Done                  ps aux | grep bash
$ echo $!
663
```

NOTE

Hãy nhớ rằng ký hiệu & được sử dụng để bắt đầu các tiến trình ở chế độ chạy ngầm.

Tham số vị trí từ \$0 đến \$9

Chúng mở rộng sang các tham số hoặc đối số được truyền cho hàm (bí danh hoặc tệp lệnh) — \$0 mở rộng sang tên của tệp lệnh hoặc vỏ.

Hãy tạo một hàm để thể hiện các tham số vị trí — lưu ý rằng PS2 (>) biểu thị các dòng mới sau dấu ngắt dòng:

```
$ special_vars() {
> echo $0
```

```
> echo $1
> echo $2
> echo $3
}
```

Bây giờ, chúng ta sẽ gọi hàm (`special_vars`) và truyền ba tham số cho nó (`debian`, `ubuntu`, `zorin`):

```
$ special_vars debian ubuntu zorin
-bash
debian
ubuntu
zorin
```

Nó đã hoạt động đúng như mong đợi.

Mặc dù việc truyền tham số vị trí cho bí danh về mặt kỹ thuật là hoàn toàn có thể nhưng nó lại hoàn toàn không có chức năng vì — với bí danh — tham số vị trí luôn được truyền ở cuối:

WARNING

```
$ alias great_editor='echo $1 is a great text editor'
$ great_editor emacs
is a great text editor emacs
```

Các biến tích hợp sẵn đặc biệt khác của Bash bao gồm:

\$#

Nó mở rộng đến số lượng đối số được truyền cho lệnh.

\$@, \$*

Chúng mở rộng đến các đối số được truyền cho lệnh.

\$_

Nó mở rộng đến tham số cuối cùng hoặc tên của tập lệnh (hoặc nhiều hơn thế nữa - hãy xem `man bash` để tìm hiểu thêm!).

Biến trong Hàm

Tất nhiên, các biến cũng có thể được sử dụng trong hàm.

Để chứng minh cho điều này, chúng ta sẽ tạo một tệp trống mới có tên `funed` và đặt hàm sau vào đó:

```
editors() {
    editor=emacs

    echo "My editor is: $editor. $editor is a fun text editor."
}
```

Như có thể đoán được, trước tiên chúng ta phải lấy nguồn của tệp để có thể gọi hàm:

```
$ . funed
```

Và bây giờ chúng ta có thể kiểm tra nó:

```
$ editors
My editor is emacs. emacs is a fun text editor.
```

Như có thể thấy, để hàm `editors` hoạt động chính xác, trước tiên biến `editor` phải được thiết lập. Phạm vi của biến đó là cục bộ trong vỏ hiện tại và chúng ta có thể tham chiếu nó trong giới hạn của phiên:

```
$ echo $editor
emacs
```

Cùng với các biến cục bộ, chúng ta cũng có thể bao gồm cả các biến môi trường trong hàm của mình:

```
editors() {
    editor=emacs

    echo "The text editor of $USER is: $editor."
}

editors
```

Hãy lưu ý lần này chúng ta đã quyết định gọi hàm từ bên trong tệp (`editors` ở dòng cuối cùng).

Bằng cách này, khi chúng ta lấy nguồn của tệp, hàm này cũng sẽ được gọi cùng lúc:

```
$ . funed
The text editor of user2 is: emacs.
```

Tham số vị trí trong Hàm

Điều tương tự cũng sẽ xảy ra với các tham số vị trí.

Chúng ta có thể truyền chúng đến các hàm từ bên trong tệp hoặc tệp lệnh (lưu ý dòng cuối cùng: `editors tortoise`):

```
editors() {
    editor=emacs

    echo "The text editor of $USER is: $editor."
    echo "Bash is not a $1 shell."
}

editors tortoise
```

Chúng ta sẽ lấy nguồn tệp và chứng minh là nó hoạt động:

```
$ . funed
The text editor of user2 is: emacs.
Bash is not a tortoise shell.
```

Và chúng ta cũng có thể truyền tham số vị trí cho các hàm tại dòng lệnh. Để chứng minh điều này, hãy loại bỏ dòng cuối cùng của tệp:

```
editors() {
    editor=emacs

    echo "The text editor of $USER is: $editor."
    echo "Bash is not a $1 shell."
}
```

Sau đó, chúng ta phải lấy nguồn tệp:

```
$ . funed
```

Cuối cùng, chúng ta gọi hàm với `tortoise` làm tham số vị trí \$1 tại dòng lệnh:

```
$ editors tortoise
The text editor of user2 is: emacs.
Bash is not a tortoise shell.
```

Hàm trong Tập lệnh

Các hàm hầu hết có thể được tìm thấy trong các tập lệnh Bash.

Việc biến tập `funed` của chúng ta thành một tập lệnh (đặt tên là `funed.sh`) là một việc rất dễ dàng:

```
#!/bin/bash

editors() {

editor=emacs

echo "The text editor of $USER is: $editor."
echo "Bash is not a $1 shell."
}

editors tortoise
```

Chính là như vậy! Chúng ta đã chỉ thêm hai dòng:

- Dòng đầu tiên là *shebang* và nó sẽ xác định chương trình nào sẽ diễn giải tập lệnh: `#!/bin/bash`. Thật kỳ lạ là bản thân chương trình đó lại chính là `bash`.
- Dòng cuối cùng chỉ đơn giản là lời gọi hàm.

Bây giờ chỉ còn một việc — chúng ta phải làm cho tập lệnh có thể thực thi được:

```
$ chmod +x funed.sh
```

Và bây giờ nó đã sẵn sàng để được thực thi:

```
$ ./funed.sh
The text editor of user2 is: emacs.
Bash is not a tortoise shell.
```

NOTE | Bạn sẽ tìm hiểu tất cả về *tập lệnh vỏ* trong các bài học tiếp theo.

Hàm trong Bí danh

Như đã nói ở trên, chúng ta có thể đặt một hàm bên trong một bí danh:

```
$ alias great_editor='gr8_ed() { echo $1 is a great text editor; unset -f gr8_ed; }; gr8_ed'
```

Giá trị bí danh dài này cần được giải thích. Hãy cùng chia nhỏ nó ra:

- Đầu tiên là hàm: `gr8_ed() { echo $1 is a great text editor; unset -f gr8_ed; }`
- Lệnh cuối cùng trong hàm — `unset -f gr8_ed` — sẽ hủy thiết lập hàm để nó không còn trong phiên `bash` hiện tại sau khi bí danh được gọi.
- Cuối cùng nhưng không kém phần quan trọng, để gọi bí danh thành công, trước tiên chúng ta cũng phải gọi hàm `gr8_ed`.

Hãy cùng gọi bí danh và chứng minh rằng nó sẽ hoạt động:

```
$ great_editor emacs
emacs is a great text editor
```

Như được thể hiện trong `unset -f gr8_ed` ở trên, lệnh `unset` không chỉ được sử dụng để hủy đặt các biến mà còn cả các hàm. Trên thực tế, chúng ta có những khoá chuyển hoặc tùy chọn cụ thể:

unset -v

dành cho biến.

unset -f

dành cho hàm.

Nếu được sử dụng mà không có khoá chuyển, `unset` sẽ cố gắng hủy đặt một biến trước tiên và — nếu thất bại — sau đó sẽ cố gắng hủy thiết lập một hàm.

Hàm trong Hàm

Bây giờ, giả sử chúng ta muốn thông báo hai điều với `user2` mỗi khi họ đăng nhập vào hệ thống:

- Chào hỏi và đề xuất/khen ngợi một trình soạn thảo văn bản.
- Vì họ đang bắt đầu đặt nhiều tệp video Matroska vào thư mục `$HOME/Video` của nó, chúng ta cần cảnh báo họ.

Để thực hiện điều đó, chúng ta đã đặt hai hàm sau vào `/home/user2/.bashrc`:

Hàm đầu tiên (`check_vids`) sẽ thực hiện kiểm tra các tệp `.mkv` và cảnh báo:

```
check_vids() {
    ls -l ~/Video/*.mkv > /dev/null 2>&1
    if [ "$?" = "0" ];then
        echo -e "Remember, you must not keep more than 5 video files in your Video
folder.\nThanks."
    else
        echo -e "You do not have any videos in the Video folder. You can keep up to 5.\nThanks."
    fi
}
```

`check_vids` sẽ thực hiện ba việc:

- Nó liệt kê các tệp `mkv` trong `~/Video` gửi đầu ra — và bất kỳ lỗi nào — đến nền tảng *bit-bucket* (`/dev/null`).
- Nó kiểm tra kết quả đầu ra của lệnh trước đó xem có thành công hay không.
- Tùy thuộc vào kết quả kiểm tra, nó sẽ lặp lại một trong hai thông báo.

Hàm thứ hai chính là một phiên bản sửa đổi của hàm `editors` của chúng ta:

```
editors() {

editor=emacs

echo "Hi, $USER!"
echo "$editor is more than a text editor!"

check_vids
}
```

```
editors
```

Quan trọng nhất là phải quan sát hai yếu tố sau:

- Lệnh cuối cùng của `editors` sẽ gọi `check_vids` để cả hai hàm xâu chuỗi với nhau: Lời chào, lời khen ngợi cũng như tác vụ kiểm tra và cảnh báo sẽ được thực hiện theo trình tự.
- Bản thân `editors` chính là điểm nhập của chuỗi hàm nên nó sẽ được gọi ở dòng cuối cùng (`editors`).

Bây giờ, hãy đăng nhập với tên `user2` và chứng minh rằng nó sẽ hoạt động:

```
# su - user2
Hi, user2!
emacs is more than a text editor!
Remember, you must not keep more than 5 video files in your Video folder.
Thanks.
```

Bài tập Hướng dẫn

1. Hãy hoàn thành bảng với “Có” hoặc “Không” khi xem xét các tính năng của bí danh và hàm:

Tính năng	Bí danh?	Hàm?
Biến cục bộ có thể được sử dụng		
Biến môi trường có thể được sử dụng		
Có thể thoát bằng \		
Có thể đệ quy		
Rất hiệu quả khi được sử dụng với các tham số vị trí		

2. Hãy nhập lệnh để liệt kê tất cả các bí danh trong hệ thống của bạn:

3. Hãy viết một bí danh có tên là `logg` để liệt kê tất cả các tệp `ogg` trong `~/Music` — mỗi tệp trên một dòng:

4. Hãy gọi bí danh để chứng minh rằng nó có hoạt động:

5. Bây giờ, hãy sửa đổi bí danh để nó hiển thị người dùng của phiên và một dấu hai chấm trước danh sách:

6. Hãy gọi lại lần nữa để chứng minh rằng phiên bản mới này cũng hoạt động:

7. Hãy liệt kê lại tất cả các bí danh và kiểm tra xem bí danh `logg` của bạn có xuất hiện trong danh sách hay không:

8. Hãy xóa bí danh:

9. Hãy xem xét các cột “Tên bí danh” và “Lệnh được đặt Bí danh `” và gán chính xác bí danh cho các giá trị của chúng:

Tên bí danh	Lệnh được đặt Bí danh	Gán bí danh
b	bash	
bash_info	which bash + echo "\$BASH_VERSION"	
kernel_info	uname -r	
greet	echo Hi, \$USER!	
computer	pc=slimbook + echo My computer is a \$pc	

10. Với tư cách là root, hãy viết một hàm có tên my_fun trong /etc/bash.bashrc. Hàm phải chào người dùng và cho họ biết đường dẫn của họ là gì. Hãy gọi nó để người dùng nhận được cả hai thông báo mỗi khi họ đăng nhập:

11. Hãy đăng nhập với tư cách là user2 để kiểm tra xem nó có hoạt động hay không:

12. Hãy viết hàm tương tự chỉ với một dòng:

13. Hãy gọi hàm:

14. Hãy huỷ thiết lập hàm:

15. Đây là phiên bản sửa đổi của hàm special_vars:

```
$ special_vars2() {
> echo $#
> echo $_
> echo $1
> echo $4
> echo $6
> echo $7
> echo $_
```



```
> echo $@
> echo $?
> }
```

Đây là lệnh chúng ta sử dụng để gọi nó:

```
$ special_vars2 crying cockles and mussels alive alive oh
```

Hãy dự đoán kết quả đầu ra:

Tham chiếu	Giá trị
echo \$#	
echo \$_	
echo \$1	
echo \$4	
echo \$6	
echo \$7	
echo \$_	
echo \$@	
echo \$?	

16. Dựa vào hàm mẫu (`check_vids`) trong phần “Hàm trong Hàm”, hãy viết một hàm có tên `check_music` để đưa vào tệp lệnh khởi động bash chấp nhận các tham số vị trí để chúng ta có thể sửa đổi nó một cách dễ dàng:

- loại tệp đang được kiểm tra: `ogg`
- thư mục lưu file: `~/Music`
- loại tệp đang được lưu giữ: `music`
- số lượng tệp đang được lưu: `7`

Bài tập Mở rộng

1. Các hàm chỉ đọc (read-only) là những hàm mà chúng ta không thể sửa đổi nội dung của nó. Hãy tìm hiểu về *hàm chỉ đọc* và hoàn thành bảng sau:

Tên hàm	Làm cho nó trở thành "chỉ đọc"	Liệt kê tất cả các hàm chỉ đọc
my_fun		

2. Hãy tìm hiểu trên mạng cách sửa đổi PS1 và bất kỳ điều gì khác mà bạn có thể cần để viết một hàm có tên là `fyi` (được đặt trong tập lệnh khởi động) cung cấp cho người dùng các thông tin sau:

- tên người dùng
- thư mục chính
- tên máy chủ
- loại hệ điều hành
- đường dẫn tìm kiếm tệp thực thi
- thư mục email
- tần suất kiểm tra email
- độ "sâu" của phiên vỏ hiện tại (bao nhiêu vỏ)
- dấu nhắc lệnh (bạn nên sửa đổi để nó hiển thị `<user>@<host-date>`)

Tóm tắt

Trong bài học này, chúng ta đã học về:

- Cả bí danh và hàm đều là những tính năng quan trọng của vỏ cho phép chúng ta gói gọn các khối mã lặp lại.
- Bí danh rất hữu ích trong việc giúp ta có được các phiên bản ngắn hơn của các lệnh dài và/hoặc phức tạp.
- Hàm là các quy trình triển khai logic và cho phép chúng ta tự động hóa các tác vụ, đặc biệt khi được sử dụng trong các tệp lệnh.
- Cú pháp viết bí danh và hàm.
- Cách nối các lệnh khác nhau bằng dấu chấm phẩy (;).
- Cách sử dụng dấu trích dẫn đúng cách với bí danh.
- Cách tạo bí danh và hàm bền vững.
- Các biến tích hợp sẵn đặc biệt trong Bash: `$?`, `$$`, `$!`, tham số vị trí (`$0-$9`), `$#`, `$@`, `$*` và `$_`.
- Cách sử dụng biến và tham số vị trí với hàm.
- Cách sử dụng các hàm trong tệp lệnh.
- Cách gọi hàm từ bí danh.
- Cách gọi hàm từ một hàm khác.
- Những khái niệm cơ bản để tạo tệp lệnh `bash`.

Các lệnh và từ khóa được sử dụng trong bài học này:

alias

Tạo bí danh.

unalias

Loại bỏ bí danh.

cd

Thay đổi thư mục.

grep

In các dòng khớp với mẫu.

function

Từ khóa vỏ để tạo hàm.

.

Lấy nguồn một tệp.

source

Lấy nguồn một tệp.

ps

Báo cáo hình ảnh tức thời của các tiến trình hiện tại.

echo

Hiển thị một dòng văn bản.

chmod

Thay đổi các bit chế độ của một tệp (ví dụ như làm cho nó có thể thực thi được).

unset

Bỏ đặt các biến và hàm.

su

Thay đổi ID người dùng hoặc trở thành siêu người dùng.

Đáp án Bài tập Hướng dẫn

1. Hãy hoàn thành bảng với “Có” hoặc “Không” khi xem xét các tính năng của bí danh và hàm:

Tính năng	Bí danh?	Hàm?
Biến cục bộ có thể được sử dụng	Có	Không
Biến môi trường có thể được sử dụng	Có	Có
Có thể thoát bằng \	Có	Không
Có thể đệ quy	Có	Có
Rất hiệu quả khi được sử dụng với các tham số vị trí	Không	Có

2. Hãy nhập lệnh để liệt kê tất cả các bí danh trong hệ thống của bạn:

```
alias
```

3. Hãy viết một bí danh có tên là `logg` để liệt kê tất cả các tệp `ogg` trong `~/Music` — mỗi tệp trên một dòng:

```
alias logg='ls -1 ~/Music/*ogg'
```

4. Hãy gọi bí danh để chứng minh rằng nó có hoạt động:

```
logg
```

5. Bây giờ, hãy sửa đổi bí danh để nó hiển thị người dùng của phiên và một dấu hai chấm trước danh sách:

```
alias logg='echo $USER;; ls -1 ~/Music/*ogg'
```

6. Hãy gọi lại lần nữa để chứng minh rằng phiên bản mới này cũng hoạt động:

```
logg
```

7. Hãy liệt kê lại tất cả các bí danh và kiểm tra xem bí danh `logg` của bạn có xuất hiện trong danh sách hay không:

```
alias
```

8. Hãy xóa bí danh:

```
unalias logg
```

9. Hãy xem xét các cột “Tên bí danh” và “Lệnh được đặt Bí danh” và gán chính xác bí danh cho các giá trị của chúng:

Tên bí danh	Lệnh được đặt Bí danh	Gán bí danh
<code>b</code>	<code>bash</code>	<code>alias b=bash</code>
<code>bash_info</code>	<code>which bash + echo "\$BASH_VERSION"</code>	<code>alias bash_info='which bash; echo "\$BASH_VERSION"'</code>
<code>kernel_info</code>	<code>uname -r</code>	<code>alias kernel_info='uname -r'</code>
<code>greet</code>	<code>echo Hi, \$USER!</code>	<code>alias greet='echo Hi, \$USER!'</code>
<code>computer</code>	<code>pc=slimbook + echo My computer is a \$pc</code>	<code>alias computer='pc=slimbook; echo My computer is a \$pc'</code>

NOTE Dấu trích dẫn đơn cũng có thể được thay thế bằng dấu trích dẫn kép.

10. Với tư cách là `root`, hãy viết một hàm có tên `my_fun` trong `/etc/bash.bashrc`. Hàm phải chào người dùng và cho họ biết đường dẫn của họ là gì. Hãy gọi nó để người dùng nhận được cả hai thông báo mỗi khi họ đăng nhập:

Lựa chọn A:

```
my_fun() {
  echo Hello, $USER!
  echo Your path is: $PATH
}
```

```
my_fun
```

Lựa chọn B:

```
function my_fun {
echo Hello, $USER!
echo Your path is: $PATH
}
my_fun
```

11. Hãy đăng nhập với tư cách là `user2` để kiểm tra xem nó có hoạt động hay không:

```
su - user2
```

12. Hãy viết hàm tương tự chỉ với một dòng:

Lựa chọn A:

```
my_fun() { echo "Hello, $USER!"; echo "Your path is: $PATH"; }
```

Lựa chọn B:

```
function my_fun { echo "Hello, $USER!"; echo "Your path is: $PATH"; }
```

13. Hãy gọi hàm:

```
my_fun
```

14. Hãy huỷ thiết lập hàm:

```
unset -f my_fun
```

15. Đây là phiên bản sửa đổi của hàm `special_vars`:

```
$ special_vars2() {
> echo $#
> echo $_
```

```
> echo $1
> echo $4
> echo $6
> echo $7
> echo $_
> echo $@
> echo $?
> }
```

Đây là lệnh chúng ta sử dụng để gọi nó:

```
$ special_vars2 crying cockles and mussels alive alive oh
```

Hãy dự đoán kết quả đầu ra:

Tham chiếu	Giá trị
echo \$#	7
echo \$_	7
echo \$1	crying
echo \$4	mussels
echo \$6	alive
echo \$7	oh
echo \$_	oh
echo \$@	crying cockles and mussels alive alive oh
echo \$?	0

16. Dựa vào hàm mẫu (`check_vids`) trong phần “Hàm trong Hàm”, hãy viết một hàm có tên `check_music` để đưa vào tập lệnh khởi động `bash` chấp nhận các tham số vị trí để chúng ta có thể sửa đổi nó một cách dễ dàng:

- loại tệp đang được kiểm tra: `ogg`
- thư mục lưu file: `~/Music`
- loại tệp đang được lưu giữ: `music`
- số lượng tệp đang được lưu: `7`


```
check_music() {
    ls -l ~/$1/*.$2 > ~/.mkv.log 2>&1
    if [ "$?" = "0" ];then
        echo -e "Remember, you must not keep more than $3 $4 files in your $1
folder.\nThanks."
    else
        echo -e "You do not have any $4 files in the $1 folder. You can keep up to
$3.\nThanks."
    fi
}

check_music Music ogg 7 music
```

Đáp án Bài tập Mở rộng

1. Các hàm chỉ đọc (read-only) là những hàm mà chúng ta không thể sửa đổi nội dung của nó. Hãy tìm hiểu về *hàm chỉ đọc* và hoàn thành bảng sau:

Tên chức năng	Làm cho nó trở thành "chỉ đọc"	Liệt kê tất cả các hàm chỉ đọc
my_fun	<code>readonly -f my_fun</code>	<code>readonly -f</code>

2. Hãy tìm hiểu trên mạng cách sửa đổi PS1 và bất kỳ điều gì khác mà bạn có thể cần để viết một hàm có tên là `fyi` (được đặt trong tập lệnh khởi động) cung cấp cho người dùng các thông tin sau:

- tên người dùng
- thư mục chính
- tên máy chủ
- loại hệ điều hành
- đường dẫn tìm kiếm tệp thực thi
- thư mục email
- tần suất kiểm tra email
- độ "sâu" của phiên vỏ hiện tại (bao nhiêu vỏ)
- dấu nhắc lệnh (bạn nên sửa đổi để nó hiển thị `<user>@<host-date>`)

```
fyi() {
    echo -e "For your Information:\n
    Username: $USER
    Home directory: $HOME
    Host: $HOSTNAME
    Operating System: $OSTYPE
    Path for executable files: $PATH
    Your mail directory is $MAIL and is searched every $MAILCHECK seconds.
    The current level of your shell is: $SHLVL"
    PS1="\u@\h-\d "
}

fyi
```



105.2 Tùy chỉnh hoặc viết các Tập lệnh đơn giản

Tham khảo các mục tiêu LPI

[LPIC-1 5.0, Exam 102, Objective 105.2](#)

Khối lượng

4

Các lĩnh vực kiến thức chính

- Sử dụng cú pháp sh tiêu chuẩn (vòng lặp, kiểm tra).
- Sử dụng phép thay thế lệnh.
- Kiểm tra các giá trị trả về thành công, thất bại hoặc các thông tin khác được cung cấp bởi lệnh.
- Thực hiện các lệnh theo chuỗi.
- Thực hiện gửi thư có điều kiện tới siêu người dùng.
- Chọn chính xác trình thông dịch tập lệnh thông qua dòng shebang (#!).
- Quản lý vị trí, quyền sở hữu, tác vụ thực thi và quyền tùy chỉnh của tập lệnh.

Sau đây là danh sách một phần các tập, thuật ngữ và tiện ích được sử dụng

- `for`
- `while`
- `test`
- `if`
- `read`
- `seq`
- `exec`

- `||`
- `&&`



105.2 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	105 Vỏ và Tập lệnh Vỏ
Mục tiêu:	105.2 Tùy chỉnh hoặc viết các Tập lệnh đơn giản
Bài:	1 trên 2

Giới thiệu

Môi trường vỏ Linux cho phép người dùng sử dụng các tệp — được gọi là *tệp lệnh* — có chứa các lệnh từ bất kỳ một chương trình có sẵn nào trong hệ thống kết hợp với các lệnh vỏ tích hợp sẵn để tự động hóa các tác vụ tùy chỉnh của người dùng và/hoặc hệ thống. Thực chất có rất nhiều tác vụ bảo trì hệ điều hành được thực hiện bằng các tệp lệnh bao gồm các chuỗi lệnh, cấu trúc quyết định và vòng lặp điều kiện. Mặc dù các tệp lệnh hầu hết đều được dành cho các tác vụ liên quan đến chính hệ điều hành nhưng chúng cũng rất hữu ích cho các tác vụ hướng tới người dùng như đổi tên tệp hàng loạt, thu thập và phân tích dữ liệu hoặc bất kỳ hoạt động dòng lệnh lặp đi lặp lại nào khác.

Các tệp lệnh chỉ là các tệp văn bản hoạt động giống như các chương trình không hơn không kém. Một chương trình thực tế - trình thông dịch - sẽ đọc và thực thi các hướng dẫn được liệt kê trong tệp lệnh. Trình thông dịch cũng có thể bắt đầu một phiên tương tác mà trong đó, các lệnh — bao gồm các tệp lệnh — sẽ được đọc và thực thi khi chúng được nhập (như trường hợp của các phiên vỏ Linux). Các tệp lệnh có thể nhóm các hướng dẫn và lệnh đó khi chúng quá phức tạp để có thể triển khai dưới dạng bí danh hoặc hàm vỏ tùy chỉnh. Hơn nữa, các tệp lệnh có thể được duy trì giống như các chương trình thông thường và vì bản chất chỉ là các tệp văn bản, chúng có thể được tạo và sửa đổi bằng bất kỳ trình soạn thảo văn bản đơn giản nào.

Cấu trúc và việc thực thi Tập lệnh

Về cơ bản, tập lệnh là một chuỗi lệnh được sắp xếp phải được thực thi bởi trình thông dịch lệnh tương ứng. Cách mỗi trình thông dịch đọc tập lệnh sẽ khác nhau và có nhiều cách khác nhau để thực hiện tác vụ này trong một phiên vỏ Bash. Trình thông dịch mặc định cho tập lệnh sẽ là trình thông dịch được chỉ định trong dòng đầu tiên của tập lệnh ngay sau các ký tự `\# !` (được gọi là *shebang*). Trong một tập lệnh có hướng dẫn về vỏ Bash, dòng đầu tiên phải là `#!/bin/bash`. Bằng cách chỉ ra dòng này, trình thông dịch cho tất cả các hướng dẫn trong tập sẽ là `/bin/bash`. Ngoại trừ dòng đầu tiên, tất cả các dòng khác bắt đầu bằng ký tự sẽ bị bỏ qua. Vì vậy, chúng có thể được sử dụng để đặt lời nhắc và chú thích. Các dòng trống cũng sẽ bị bỏ qua. Do đó, một tập lệnh vỏ ngắn gọn có thể được viết như sau:

```
#!/bin/bash

# A very simple script

echo "Cheers from the script file! Current time is: "

date +%H:%M
```

Tập lệnh này chỉ có hai hướng dẫn cho trình thông dịch `/bin/bash`: lệnh tích hợp sẵn `echo` và lệnh `date`. Cách cơ bản nhất để chạy tập lệnh là thực thi trình thông dịch với đường dẫn tập lệnh làm đối số. Vì vậy, giả sử ví dụ trước được lưu trong tập lệnh có tên `script.sh` trong thư mục hiện tại, nó sẽ được Bash đọc và diễn giải bằng lệnh sau:

```
$ bash script.sh
Cheers from the script file! Current time is:
10:57
```

Lệnh `echo` sẽ tự động thêm một dòng mới sau khi hiển thị nội dung, nhưng tùy chọn `-n` sẽ ngăn chặn hành vi này. Do đó, việc sử dụng `echo -n` trong tập lệnh sẽ làm cho đầu ra của cả hai lệnh xuất hiện trên cùng một dòng:

```
$ bash script.sh
Cheers from the script file! Current time is: 10:57
```

Mặc dù không bắt buộc nhưng hậu tố `.sh` sẽ giúp xác định các tập lệnh vỏ khi liệt kê và tìm kiếm tập.

TIP

Bash sẽ gọi bất kỳ lệnh nào được chỉ định sau `#!` làm trình thông dịch cho tập lệnh. Ví dụ: việc sử dụng Shebang cho các ngôn ngữ tập lệnh khác như *Python* (`#!/usr/bin/python`), *Perl* (`#!/usr/bin/Perl`) hoặc *awk* (`#!/usr/bin/awk`) có thể sẽ rất hữu ích.

Nếu tập lệnh được dự định để những người dùng khác trong hệ thống thực thi, quan trọng nhất là ta phải kiểm tra xem quyền đọc thích hợp có được đặt hay không. Lệnh `chmod o+r script.sh` sẽ cấp quyền đọc cho tất cả người dùng trong hệ thống và cho phép họ thực thi `script.sh` bằng cách đặt đường dẫn đến tập lệnh làm đối số của lệnh `bash`. Ngoài ra, tập lệnh có thể được đặt quyền thực thi bit để tập có thể được thực thi như một lệnh thông thường. Bit thực thi có thể được kích hoạt trên tập lệnh bằng lệnh `chmod`:

```
$ chmod +x script.sh
```

Khi bit thực thi được bật, tập lệnh có tên `script.sh` trong thư mục hiện tại có thể được thực thi trực tiếp bằng lệnh `./script.sh`. Các tập lệnh được đặt trong thư mục được liệt kê trong biến môi trường `PATH` cũng sẽ có thể được truy cập mà không cần đường dẫn đầy đủ của chúng.

WARNING

Một tập lệnh thực hiện các hành động bị hạn chế có thể được kích hoạt quyền SUID để người dùng thông thường cũng có thể chạy nó với quyền gốc. Trong trường hợp này, điều rất quan trọng là phải đảm bảo rằng không có người dùng nào ngoài siêu người dùng có quyền ghi vào tập. Nếu không, người dùng thông thường có thể sửa đổi tập để thực hiện các hoạt động tùy ý và sẽ có khả năng gây hại.

Việc sắp xếp và thụt lề các lệnh trong tập lệnh sẽ không quá cứng nhắc. Mỗi dòng trong một tập lệnh vỏ sẽ được thực thi như một lệnh vỏ thông thường theo cùng trình tự xuất hiện của các dòng trong tập lệnh. Các quy tắc áp dụng cho dấu nhắc lệnh vỏ tương tự cũng sẽ áp dụng cho từng dòng tập lệnh riêng lẻ. Ta có thể đặt hai hoặc nhiều lệnh trên cùng một dòng và phân tách bằng dấu chấm phẩy:

```
echo "Cheers from the script file! Current time is:" ; date +%H:%M
```

Mặc dù đôi khi định dạng này có thể sẽ thuận tiện hơn nhưng việc sử dụng nó là tùy chọn vì các lệnh tuần tự có thể được đặt mỗi dòng một lệnh và chúng sẽ được thực thi theo như cách chúng được phân tách. Nói cách khác, dấu chấm phẩy có thể được thay thế bằng một ký tự dòng mới trong tập lệnh Bash.

Khi một tập lệnh được thực thi, các lệnh ở trong nó sẽ không được thực thi trực tiếp trong phiên hiện tại mà thay vào đó là bởi một tiến trình Bash mới được gọi là *vỏ con*. Nó sẽ ngăn tập lệnh ghi

đề các biến môi trường của phiên hiện tại và để lại các sửa đổi không được giám sát trong phiên hiện tại. Nếu mục tiêu của người dùng là chạy nội dung của tập lệnh trong phiên vỏ hiện tại thì nó phải được thực thi bằng `source script.sh` hoặc `. script.sh` (hãy lưu ý rằng có một khoảng cách giữa dấu chấm và tên tập lệnh).

Giống như khi thực thi bất kỳ một lệnh nào khác, dấu nhắc lệnh vỏ sẽ chỉ khả dụng trở lại khi tập lệnh kết thúc việc thực thi và mã trạng thái thoát của nó sẽ có sẵn trong biến `$?`. Để thay đổi hành vi này để vỏ hiện tại cũng kết thúc khi tập lệnh kết thúc, tập lệnh — hoặc bất kỳ một lệnh nào khác — có thể được đặt trước lệnh `exec`. Lệnh này cũng sẽ thay thế mã trạng thái thoát của phiên vỏ hiện tại bằng mã riêng của nó.

Biến

Biến trong tập lệnh vỏ hoạt động giống như trong các phiên tương tác với điều kiện là trình thông dịch phải giống nhau. Ví dụ: định dạng `SOLUTION=42` (không có khoảng trắng xung quanh dấu bằng) sẽ gán giá trị 42 cho biến có tên là `SOLUTION`. Theo quy ước, chữ in hoa sẽ được sử dụng cho tên biến (nhưng không bắt buộc). Tuy nhiên, tên biến không thể bắt đầu bằng các ký tự không nằm trong bảng chữ cái.

Ngoài các biến thông thường do người dùng tạo, tập lệnh Bash còn có một tập hợp các biến đặc biệt được gọi là *tham số*. Không giống như các biến thông thường, tên tham số sẽ bắt đầu bằng ký tự không phải chữ cái để chỉ định chức năng của nó. Các đối số được truyền cho tập lệnh và các thông tin hữu ích khác được lưu trữ trong các tham số như `$0`, `$*`, `$?`, v.v. mà trong đó, ký tự theo sau ký hiệu đô la sẽ biểu thị thông tin cần tìm nạp:

`$*`

Tất cả các đối số được truyền cho tập lệnh.

`$@`

Tất cả các đối số được truyền cho tập lệnh. Nếu được sử dụng với dấu trích dẫn kép (như trong `"$@"`), mọi đối số sẽ được đặt trong dấu trích dẫn kép.

`$#`

Số lượng đối số.

`$0`

Tên của tập lệnh.

`$!`

PID của chương trình được thực hiện cuối cùng.

\$\$

PID của vỏ hiện tại.

\$?

Mã trạng thái thoát bằng số của lệnh được hoàn thành cuối cùng. Đối với các tiến trình tiêu chuẩn POSIX, giá trị số 0 có nghĩa là lệnh cuối cùng đã được thực thi thành công; điều này cũng áp dụng cho các tập lệnh vỏ.

Một *tham số vị trí* là một tham số được biểu thị bằng một hoặc nhiều chữ số khác 0. Ví dụ: biến \$1 tương ứng với đối số đầu tiên được cung cấp cho tập lệnh (tham số vị trí một), \$2 tương ứng với đối số thứ hai, v.v. Nếu vị trí của một tham số lớn hơn chín thì tham số đó phải được tham chiếu bằng dấu ngoặc nhọn như trong \${10}, \${11}, v.v.

Mặt khác, các biến thông thường được nhắm tới mục đích lưu trữ các giá trị được chèn thủ công hoặc đầu ra được tạo bởi các lệnh khác. Ví dụ: lệnh `read` có thể được sử dụng bên trong tập lệnh để yêu cầu người dùng nhập thông tin trong quá trình thực thi tập lệnh:

```
echo "Do you want to continue (y/n)?"
read ANSWER
```

Giá trị trả về sẽ được lưu trữ trong biến ANSWER. Nếu tên biến không được cung cấp, tên biến REPLY sẽ được sử dụng theo mặc định. Ta cũng có thể sử dụng lệnh `read` để đọc nhiều biến cùng một lúc:

```
echo "Type your first name and last name:"
read NAME SURNAME
```

Trong trường hợp này, mỗi số hạng được phân tách bằng dấu cách sẽ được gán tương ứng cho các biến NAME và SURNAME. Nếu số của các số hạng đã cho lớn hơn số biến thì chúng sẽ được lưu vào biến cuối cùng. Bản thân `read` có thể hiển thị thông báo cho người dùng với tùy chọn `-p` để khiến lệnh `echo` trở nên không cần thiết trong trường hợp này:

```
read -p "Type your first name and last name:" NAME SURNAME
```

Các tập lệnh thực hiện tác vụ hệ thống thường sẽ yêu cầu thông tin do các chương trình khác cung cấp. Ký hiệu *trích dẫn đơn ngược* (```) có thể được sử dụng để lưu trữ kết quả đầu ra của lệnh trong một biến:

```
$ OS=`uname -o`
```

Trong ví dụ, đầu ra của lệnh `uname -o` sẽ được lưu trong biến `OS`. Một kết quả giống hệt sẽ được tạo ra với `$()`:

```
$ OS=$(uname -o)
```

Độ dài của một biến (tức là số lượng ký tự mà nó chứa) sẽ được trả về bằng cách thêm một hàm băm `#` vào trước tên của biến. Tuy nhiên, tính năng này yêu cầu sử dụng cú pháp dấu ngoặc nhọn để biểu thị biến:

```
$ OS=$(uname -o)
$ echo $OS
GNU/Linux
$ echo ${#OS}
9
```

Bash cũng có các biến mảng một chiều. Do đó, một tập hợp các phần tử liên quan có thể được lưu trữ bằng một tên biến duy nhất. Mỗi phần tử trong mảng đều có một chỉ mục bằng số. Chỉ số này phải được sử dụng để ghi và đọc các giá trị trong phần tử tương ứng. Khác với các biến thông thường, mảng phải được khai báo bằng lệnh `declare` tích hợp sẵn của Bash. Ví dụ: để khai báo một biến có tên `SIZES` dưới dạng một mảng:

```
$ declare -a SIZES
```

Mảng cũng có thể được khai báo ngầm khi điền từ danh sách các mục được xác định trước bằng cách sử dụng ký hiệu dấu ngoặc đơn:

```
$ SIZES=( 1048576 1073741824 )
```

Trong ví dụ này, hai giá trị số nguyên lớn đã được lưu trữ trong mảng `SIZES`. Các phần tử mảng phải được tham chiếu bằng dấu ngoặc nhọn và ngoặc vuông. Nếu không, Bash sẽ không thay đổi hoặc hiển thị phần tử một cách chính xác. Vì các chỉ mục mảng bắt đầu từ 0 nên nội dung của phần tử đầu tiên sẽ nằm trong `${SIZES[0]}`, phần tử thứ hai sẽ nằm trong `${SIZES[1]}`, v.v.

```
$ echo ${SIZES[0]}
1048576
```

```
$ echo ${SIZES[1]}
1073741824
```

Không giống như việc đọc, việc thay đổi nội dung của phần tử mảng có thể được thực hiện mà không cần dấu ngoặc nhọn (ví dụ: `SIZES[0]=1048576`). Giống như các biến thông thường, độ dài của một phần tử trong một mảng sẽ được trả về bằng ký tự băm (ví dụ: `${#SIZES[0]}` cho độ dài của phần tử đầu tiên trong mảng `SIZES`). Tổng số phần tử trong một mảng sẽ được trả về nếu `@` hoặc `*` được sử dụng làm chỉ mục:

```
$ echo ${#SIZES[@]}
2
$ echo ${#SIZES[*]}
2
```

Mảng cũng có thể được khai báo bằng cách sử dụng đầu ra của lệnh làm phần tử ban đầu thông qua thay thế lệnh. Ví dụ sau đây sẽ cho thấy cách tạo một mảng Bash có các phần tử là hệ thống tệp được hỗ trợ của hệ thống hiện tại:

```
$ FS=( $(cut -f 2 < /proc/filesystems) )
```

Lệnh `cut -f 2 < /proc/filesystems` sẽ hiển thị tất cả các hệ thống tệp hiện được hỗ trợ bởi hạt nhân đang chạy (như được liệt kê trong cột thứ hai của tệp `/proc/filesystems`). Vì vậy, mảng `FS` hiện sẽ chứa một phần tử cho mỗi một hệ thống tệp được hỗ trợ. Bất kỳ nội dung văn bản nào cũng có thể được sử dụng để khởi tạo một mảng vì theo mặc định, mọi thuật ngữ được phân cách bằng các ký tự *dấu cách*, *tab* hoặc *dấu xuống dòng* sẽ trở thành một phần tử mảng.

TIP

Bash coi mỗi ký tự của `$IFS` (*Dấu Phân cách Trường Đầu vào*) của biến môi trường là một dấu phân cách. Ví dụ: để thay đổi dấu phân cách trường thành các ký tự dòng mới, biến `IFS` phải được đặt lại bằng lệnh `IFS=$'\n'`.

Biểu thức Số học

Bash có cung cấp một phương pháp thực tế để thực hiện các phép tính số học số nguyên bằng lệnh tích hợp sẵn `expr`. Ví dụ: hai biến số `$VAL1` và `$VAL2` có thể được cộng lại bằng lệnh sau:

```
$ SUM=`expr $VAL1 + $VAL2`
```

Giá trị kết quả của ví dụ sẽ có sẵn trong biến `$SUM`. Lệnh `expr` có thể được thay thế bằng `$(())`. Vì vậy, ví dụ trước có thể được viết lại thành `SUM=$(($VAL1 + $VAL2))`. Biểu thức lũy thừa cũng

được cho phép với toán tử dấu hoa thị kép. Vì vậy, phần khai báo mảng trước đó `SIZES=(1048576 1073741824)` có thể được viết lại thành `SIZES=($(1024**2) $(1024**3))`.

Thay thế lệnh cũng có thể được sử dụng trong các biểu thức số học. Ví dụ: tệp `/proc/meminfo` có thông tin chi tiết về bộ nhớ hệ thống bao gồm số byte trống trong RAM:

```
$ FREE=$(( 1000 * `sed -nre '2s/[^[[:digit:]]//gp' < /proc/meminfo` ))
```

Ví dụ này cho thấy lệnh `sed` có thể được sử dụng như thế nào để phân tích nội dung của `/proc/meminfo` bên trong biểu thức số học. Dòng thứ hai của tệp `/proc/meminfo` có chứa lượng bộ nhớ trống tính bằng hàng nghìn byte. Do đó, biểu thức số học sẽ nhân nó với 1000 để có được số byte trống trong RAM.

Thực thi có điều kiện

Một số tệp lệnh thường không nhằm mục đích thực thi tất cả các lệnh trong tệp lệnh mà chỉ những lệnh phù hợp với tiêu chí đã được xác định trước. Ví dụ: một tệp lệnh bảo trì chỉ có thể gửi thông báo cảnh báo đến email của quản trị viên nếu việc thực thi lệnh không thành công. Bash có cung cấp các phương pháp cụ thể để đánh giá sự thành công của việc thực thi lệnh và các cấu trúc điều kiện chung tương tự như các phương pháp được tìm thấy trong các ngôn ngữ lập trình phổ biến.

Bằng cách tách các lệnh với `&&`, lệnh bên phải sẽ chỉ được thực thi nếu lệnh bên trái không gặp lỗi, nghĩa là nếu trạng thái thoát của nó bằng 0:

```
COMMAND A && COMMAND B && COMMAND C
```

Hành vi ngược lại sẽ xảy ra nếu các lệnh được phân tách bằng `||`. Trong trường hợp này, lệnh sau sẽ chỉ được thực thi nếu lệnh trước đó gặp lỗi, nghĩa là nếu mã trạng thái trả về của nó khác 0.

Một trong những tính năng quan trọng nhất của tất cả các ngôn ngữ lập trình là khả năng thực thi các lệnh tùy thuộc vào các điều kiện đã xác định trước đó. Cách đơn giản nhất để thực thi các lệnh có điều kiện là sử dụng lệnh `if` tích hợp sẵn của Bash. Lệnh này sẽ chỉ thực thi một hoặc nhiều lệnh nếu lệnh được đưa ra làm đối số trả về mã trạng thái 0 (thành công). Một lệnh khác là `test` có thể được sử dụng để đánh giá nhiều tiêu chí đặc biệt khác nhau, vì thế mà nó chủ yếu được sử dụng cùng với `if`. Trong ví dụ sau, thông báo `Confirmed: /bin/bash is executable.` sẽ được hiển thị nếu tệp `/bin/bash` tồn tại và nó có thể thực thi được:

```
if test -x /bin/bash ; then
  echo "Confirmed: /bin/bash is executable."
```

```
fi
```

Tùy chọn `-x` sẽ khiến lệnh `test` chỉ trả về mã trạng thái 0 nếu đường dẫn đã cho là một tệp thực thi. Vì dấu ngoặc vuông có thể được sử dụng để thay thế cho `test`, ví dụ sau đây sẽ cho thấy một cách khác để đạt được kết quả tương tự:

```
if [ -x /bin/bash ] ; then
    echo "Confirmed: /bin/bash is executable."
fi
```

Lệnh `else` là tùy chọn đối với cấu trúc `if` và nếu được sử dụng, nó có thể xác định một lệnh hoặc chuỗi lệnh để thực thi nếu biểu thức điều kiện không đúng:

```
if [ -x /bin/bash ] ; then
    echo "Confirmed: /bin/bash is executable."
else
    echo "No, /bin/bash is not executable."
fi
```

Cấu trúc `if` phải luôn được kết thúc bằng `fi` để trình thông dịch Bash biết nơi các lệnh điều kiện kết thúc.

Đầu ra của Tập lệnh

Ngay cả khi mục đích của tập lệnh chỉ liên quan đến các thao tác hướng đến tệp, quan trọng nhất là ta phải hiển thị các thông báo liên quan đến tiến trình ở đầu ra tiêu chuẩn để người dùng luôn được thông báo về mọi vấn đề và cuối cùng có thể sử dụng các thông báo đó để tạo nhật ký vận hành.

Lệnh tích hợp sẵn của Bash là `echo` thường được sử dụng để hiển thị các chuỗi văn bản đơn giản nhưng nó cũng cung cấp một số tính năng mở rộng. Với tùy chọn `-e`, lệnh `echo` có thể hiển thị các ký tự đặc biệt bằng cách sử dụng các chuỗi thoát (một chuỗi dấu gạch chéo ngược chỉ định một ký tự đặc biệt). Ví dụ:

```
#!/bin/bash

# Get the operating system's generic name
OS=$(uname -o)

# Get the amount of free memory in bytes
```

```
FREE=$(( 1000 * `sed -nre '2s/^[[:digit:]]//gp' < /proc/meminfo` ))

echo -e "Operating system:\t$OS"
echo -e "Unallocated RAM:\t$( ($FREE / 1024**2 )) MB"
```

Mặc dù việc sử dụng dấu trích dẫn là tùy chọn khi sử dụng lệnh `echo` mà không có tùy chọn, ta sẽ phải thêm chúng khi sử dụng tùy chọn `-e`. Nếu không, các ký tự đặc biệt có thể sẽ không hiển thị chính xác. Trong tệp lệnh trước, cả hai lệnh `echo` đều sử dụng ký tự lập bảng `\t` để căn chỉnh văn bản và dẫn đến kết quả đầu ra sau:

```
Operating system:      GNU/Linux
Unallocated RAM:      1491 MB
```

Ký tự dòng mới `\n` có thể được sử dụng để phân tách các dòng đầu ra, từ đó ta có thể có được kết quả đầu ra giống hệt nhau bằng cách kết hợp hai lệnh `echo` thành một:

```
echo -e "Operating system:\t$OS\nUnallocated RAM:\t$( ($FREE / 1024**2 )) MB"
```

Mặc dù có thể hiển thị hầu hết các thông báo văn bản, lệnh `echo` có thể sẽ không phù hợp lắm cho việc hiển thị các mẫu văn bản cụ thể hơn. Lệnh tích hợp sẵn `printf` cung cấp nhiều quyền kiểm soát hơn về cách hiển thị các biến. Lệnh `printf` sử dụng đối số đầu tiên làm định dạng đầu ra mà trong đó, các ký tự giữ chỗ sẽ được thay thế bằng các đối số sau theo thứ tự chúng xuất hiện trong dòng lệnh. Ví dụ: thông báo của ví dụ trước có thể được tạo bằng lệnh `printf` sau:

```
printf "Operating system:\t%s\nUnallocated RAM:\t%d MB\n" $OS $( ($FREE / 1024**2 ))
```

Ký tự giữ chỗ `%s` được dành cho nội dung văn bản (nó sẽ được thay thế bằng biến `$OS`) và ký tự giữ chỗ `%d` được dành cho số nguyên (nó sẽ được thay thế bằng số megabyte tổng trong RAM). Lệnh `printf` sẽ không thêm ký tự dòng mới vào cuối văn bản, vì thế mà ký tự dòng mới `\n` nên được đặt ở cuối mẫu nếu cần. Toàn bộ mẫu phải được hiểu là một đối số duy nhất. Vì vậy, nó phải được đặt trong dấu trích dẫn kép.

TIP

Định dạng của ký tự giữ chỗ thay thế được thực hiện bởi `printf` có thể được tùy chỉnh bằng cách sử dụng cùng định dạng được sử dụng bởi hàm `printf` trong ngôn ngữ lập trình C. Bạn có thể tìm thấy tham chiếu đầy đủ về hàm `printf` trong trang hướng dẫn của nó (được truy cập bằng lệnh `man 3 printf`).

Với `printf`, các biến sẽ được đặt bên ngoài mẫu văn bản. Điều này giúp chúng ta lưu trữ mẫu văn bản trong một biến riêng biệt:

```
MSG='Operating system:\t%s\nUnallocated RAM:\t%d MB\n'  
printf "$MSG" $OS $(( $FREE / 1024**2 ))
```

Phương pháp này đặc biệt hữu ích trong việc hiển thị các định dạng đầu ra riêng biệt, tùy thuộc vào yêu cầu của người dùng. Ví dụ: sẽ dễ dàng hơn khi ta viết tập lệnh sử dụng mẫu văn bản riêng biệt nếu người dùng yêu cầu danh sách CSV (*Giá trị được phân tách bằng dấu phẩy*) thay vì thông báo đầu ra mặc định.

Bài tập Hướng dẫn

1. Tùy chọn `-s` cho lệnh `read` rất hữu ích cho việc nhập mật khẩu vì nó sẽ không hiển thị nội dung được nhập trên màn hình. Làm thế nào để có thể sử dụng lệnh `read` để lưu trữ dữ liệu đầu vào của người dùng trong biến `PASSWORD` trong khi ẩn nội dung đã nhập?

2. Mục đích duy nhất của lệnh `whoami` là để hiển thị tên người dùng đã gọi nó. Vì vậy, nó chủ yếu được sử dụng bên trong các tập lệnh để xác định người dùng đang chạy tập lệnh. Bên trong tập lệnh Bash, làm thế nào để đầu ra của lệnh `whoami` có thể được lưu trữ trong biến có tên `WHO`?

3. Toán tử Bash nào nên nằm giữa các lệnh `apt-get dist-upgrade` và `systemctl restart` nếu siêu người dùng (root) chỉ muốn thực thi `systemctl restart` trong trường hợp `apt-get dist-upgrade` hoàn tất thành công?

Bài tập Mở rộng

1. Sau khi thử chạy tập lệnh Bash mới được tạo, người dùng nhận được thông báo lỗi sau:

```
bash: ./script.sh: Permission denied
```

Lưu ý rằng tập `./script.sh` được tạo bởi cùng một người dùng, nguyên nhân có thể gây ra lỗi này là gì?

2. Giả sử một tập lệnh có tên `do.sh` có thể thực thi được và liên kết tượng trưng có tên `undo.sh` trỏ đến nó. Từ trong tập lệnh, làm cách nào để có thể xác định tên tập đang gọi là `do.sh` hay `undo.sh`?

3. Trong hệ thống có một dịch vụ email được cấu hình đúng cách, lệnh `mail -s "Maintenance Error" root <<<"Scheduled task error"` sẽ gửi thông báo email đến siêu người dùng. Lệnh như vậy có thể được sử dụng trong các tác vụ không được giám sát (như *các công việc định kỳ*) để thông báo cho quản trị viên hệ thống về một sự cố không mong muốn. Hãy viết cấu trúc `if` sẽ thực thi lệnh `mail` đã nói ở trên nếu trạng thái thoát của lệnh trước đó — bất kể đó là gì — là không thành công.

Tóm tắt

Bài học này bao gồm các khái niệm cơ bản để hiểu và viết các tập lệnh vỏ Bash. Các tập lệnh Vỏ là một phần cốt lõi của bất kỳ bản phân phối Linux nào vì chúng cung cấp một cách rất linh hoạt để tự động hóa các tác vụ hệ thống và người dùng được thực hiện trong môi trường vỏ. Bài học đã đi qua các bước sau:

- Cấu trúc tập lệnh Vỏ và quyền truy cập tập lệnh chính xác
- Thông số tập lệnh
- Sử dụng các biến để đọc đầu vào của người dùng và lưu trữ đầu ra của lệnh
- Mảng Bash
- Kiểm tra đơn giản và thực thi có điều kiện
- Định dạng đầu ra

Các lệnh và quy trình đã được nhắc đến là:

- Ký hiệu tích hợp sẵn của Bash để thay thế lệnh, mở rộng mảng và biểu thức số học
- Thực thi lệnh có điều kiện với toán tử `||` và `&&`
- `echo`
- `chmod`
- `exec`
- `read`
- `declare`
- `test`
- `if`
- `printf`

Đáp án Bài tập Hướng dẫn

1. Tùy chọn `-s` cho lệnh `read` rất hữu ích cho việc nhập mật khẩu vì nó sẽ không hiển thị nội dung được nhập trên màn hình. Làm thế nào để có thể sử dụng lệnh `read` để lưu trữ dữ liệu đầu vào của người dùng trong biến `PASSWORD` trong khi ẩn nội dung đã nhập?

```
read -s PASSWORD
```

2. Mục đích duy nhất của lệnh `whoami` là để hiển thị tên người dùng đã gọi nó. Vì vậy, nó chủ yếu được sử dụng bên trong các tệp lệnh để xác định người dùng đang chạy tệp lệnh. Bên trong tệp lệnh Bash, làm thế nào để đầu ra của lệnh `whoami` có thể được lưu trữ trong biến có tên `WHO`?

```
WHO=`whoami` or WHO=$(whoami)
```

3. Toán tử Bash nào nên nằm giữa các lệnh `apt-get dist-upgrade` và `systemctl restart` nếu siêu người dùng (root) chỉ muốn thực thi `systemctl restart` trong trường hợp `apt-get dist-upgrade` hoàn tất thành công?

Toán tử `&&`, như trong `apt-get dist-upgrade && systemctl restart`.

Đáp án Bài tập Mở rộng

1. Sau khi thử chạy tập lệnh Bash mới được tạo, người dùng nhận được thông báo lỗi sau:

```
bash: ./script.sh: Permission denied
```

Lưu ý rằng tập `./script.sh` được tạo bởi cùng một người dùng, nguyên nhân có thể gây ra lỗi này là gì?

Tập `./script.sh` chưa được gán quyền thực thi.

2. Giả sử một tập lệnh có tên `do.sh` có thể thực thi được và liên kết tượng trưng có tên `undo.sh` trỏ đến nó. Từ trong tập lệnh, làm cách nào để có thể xác định tên tập đang gọi là `do.sh` hay `undo.sh`?

Biến đặc biệt `$0` chứa tên tập được sử dụng để gọi tập lệnh.

3. Trong hệ thống có một dịch vụ email được cấu hình đúng cách, lệnh `mail -s "Maintenance Error" root <<<"Scheduled task error"` sẽ gửi thông báo email đến siêu người dùng. Lệnh như vậy có thể được sử dụng trong các tác vụ không được giám sát (như *các công việc định kỳ*) để thông báo cho quản trị viên hệ thống về một sự cố không mong muốn. Hãy viết cấu trúc `if` sẽ thực thi lệnh `mail` đã nói ở trên nếu trạng thái thoát của lệnh trước đó — bất kể đó là gì — là không thành công.

```
if [ "$?" -ne 0 ]; then mail -s "Maintenance Error" root <<<"Scheduled task error"; fi
```



105.2 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	105 Vỏ và Tập lệnh Vỏ
Mục tiêu:	105.2 Tùy chỉnh hoặc viết các Tập lệnh đơn giản
Bài:	2 trên 2

Giới thiệu

Tập lệnh Vỏ thường được sử dụng nhằm mục đích tự động hóa các hoạt động liên quan đến tệp và thư mục - chính là các hoạt động có thể được thực hiện thủ công tại dòng lệnh. Tuy nhiên, phạm vi hoạt động của các tập lệnh vỏ không bị giới hạn ở tài liệu của người dùng vì việc cấu hình và tương tác với nhiều khía cạnh của hệ điều hành Linux cũng được thực hiện thông qua các tập lệnh.

Vỏ Bash có cung cấp nhiều lệnh tích hợp sẵn hữu ích để viết các tập lệnh vỏ, nhưng toàn bộ sức mạnh của các tập lệnh này phụ thuộc vào sự kết hợp của các lệnh tích hợp sẵn của Bash với các tiện ích dòng lệnh có sẵn trên hệ thống Linux.

Kiểm tra mở rộng

Bash là một ngôn ngữ tập lệnh chủ yếu được định hướng để làm việc với các tệp. Vì vậy, lệnh `test` tích hợp sẵn của Bash có nhiều tùy chọn để đánh giá đặc tính của các đối tượng hệ thống tệp (về cơ bản là tệp và thư mục). Các phiên kiểm tra tập trung vào tệp và thư mục thực sự rất hữu ích: chẳng hạn như để xác minh xem các tệp và thư mục cần thiết để thực hiện một tác vụ cụ thể có hiện diện và có thể đọc được hay không. Sau đó, cùng với cấu trúc có điều kiện `if`, một chuỗi hành

động thích hợp sẽ được thực thi nếu kiểm tra thành công.

Lệnh `test` có thể đánh giá các biểu thức bằng hai cú pháp khác nhau: các biểu thức kiểm tra có thể được đưa ra làm đối số cho lệnh `test` hoặc có thể được đặt bên trong dấu ngoặc vuông (lệnh `test` được thực hiện ngầm). Do đó, việc kiểm tra để đánh giá xem `/etc` có phải là một thư mục hợp lệ hay không có thể được viết dưới dạng `test -d /etc` hoặc `[-d /etc]`:

```
$ test -d /etc
$ echo $?
0
$ [ -d /etc ]
$ echo $?
0
```

Như được xác nhận bởi mã trạng thái thoát trong biến đặc biệt `$?` — một giá trị 0 có nghĩa là đã kiểm tra thành công — cả hai mẫu đều đánh giá `/etc` là một thư mục hợp lệ. Giả sử đường dẫn đến một tệp hoặc thư mục được lưu trữ trong biến `$VAR`, các biểu thức sau có thể được sử dụng làm đối số cho `test` hoặc bên trong dấu ngoặc vuông:

-a "\$VAR"

Đánh giá xem đường dẫn trong `VAR` có tồn tại trong hệ thống tệp hay không và đó có phải là một tệp hay không.

-b "\$VAR"

Đánh giá xem đường dẫn trong `VAR` có phải là một tệp khối đặc biệt hay không.

-c "\$VAR"

Đánh giá xem đường dẫn trong `VAR` có phải là một tệp ký tự đặc biệt hay không.

-d "\$VAR"

Đánh giá xem đường dẫn trong `VAR` có phải là một thư mục hay không.

-e "\$VAR"

Đánh giá xem đường dẫn trong `VAR` có tồn tại trong hệ thống tệp hay không.

-f "\$VAR"

Đánh giá xem đường dẫn trong `VAR` có tồn tại hay không và đó có phải là một tệp thông thường hay không.

-g "\$VAR"

Đánh giá xem đường dẫn trong `VAR` có quyền SGID hay không.

-h "\$VAR"

Đánh giá xem đường dẫn trong VAR có phải là liên kết tượng trưng hay không.

-L "\$VAR"

Đánh giá xem đường dẫn trong VAR có phải là liên kết tượng trưng hay không (như -h).

-k "\$VAR"

Đánh giá xem đường dẫn trong VAR có quyền bit *dính* hay không.

-p "\$VAR"

Đánh giá xem đường dẫn trong VAR có phải là tệp *ống* hay không.

-r "\$VAR"

Đánh giá xem người dùng hiện tại có thể đọc được đường dẫn trong VAR hay không.

-s "\$VAR"

Đánh giá xem đường dẫn trong VAR có tồn tại và nó có trống hay không.

-S "\$VAR"

Đánh giá xem đường dẫn trong VAR có phải là tệp ổ nổi hay không.

-t "\$VAR"

Đánh giá xem đường dẫn trong VAR có mở trong một cửa sổ dòng lệnh hay không.

-u "\$VAR"

Đánh giá xem đường dẫn trong VAR có quyền SUID hay không.

-w "\$VAR"

Đánh giá xem người dùng hiện tại có thể ghi đường dẫn trong VAR hay không.

-x "\$VAR"

Đánh giá xem người dùng hiện tại có thể thực thi được đường dẫn trong VAR hay không.

-O "\$VAR"

Đánh giá xem đường dẫn trong VAR có thuộc quyền sở hữu của người dùng hiện tại hay không.

-G "\$VAR"

Đánh giá xem đường dẫn trong VAR có thuộc nhóm có hiệu lực của người dùng hiện tại hay không.

-N "\$VAR"

Đánh giá xem đường dẫn trong VAR có được sửa đổi kể từ lần cuối nó được truy cập hay không.

"\$VAR1" -nt "\$VAR2"

Đánh giá xem đường dẫn trong VAR1 có mới hơn đường dẫn trong VAR2 hay không (theo ngày sửa đổi của chúng).

"\$VAR1" -ot "\$VAR2"

Đánh giá xem đường dẫn trong VAR1 có cũ hơn VAR2 hay không.

"\$VAR1" -ef "\$VAR2"

Đánh giá là Đúng nếu đường dẫn trong VAR1 là liên kết cứng đến VAR2.

Chúng ta nên sử dụng dấu trích dẫn kép xung quanh một biến được kiểm tra vì nếu biến đó trống thì nó có thể gây ra lỗi cú pháp cho lệnh test. Các tùy chọn kiểm tra sẽ yêu cầu một đối số toán hạng và một biến trống không được trích dẫn sẽ gây ra lỗi do thiếu đối số bắt buộc. Ngoài ra, chúng ta còn có những phiên kiểm tra các biến văn bản tùy ý được mô tả như sau:

-z "\$TXT"

Đánh giá xem biến TXT có trống không (kích thước bằng 0)

-n "\$TXT" hoặc test "\$TXT"

Đánh giá xem biến TXT có trống không.

"\$TXT1" = "\$TXT2" hoặc "\$TXT1" == "\$TXT2"

Đánh giá xem TXT1 và TXT2 có bằng nhau không.

"\$TXT1" != "\$TXT2"

Đánh giá xem TXT1 và TXT2 có không bằng nhau hay không.

"\$TXT1" < "\$TXT2"

Đánh giá xem TXT1 có đứng trước TXT2 hay không, theo thứ tự bảng chữ cái.

"\$TXT1" > "\$TXT2"

Đánh giá xem TXT1 có đứng sau TXT2 hay không, theo thứ tự bảng chữ cái.

Các ngôn ngữ khác nhau có thể có các quy tắc khác nhau về thứ tự bảng chữ cái. Để có được kết quả nhất quán bất kể cài đặt cục bộ của hệ thống nơi tập lệnh đang được thực thi có như thế nào, chúng ta nên đặt biến môi trường LANG thành C như trong LANG=C trước khi thực hiện các thao tác liên quan đến thứ tự bảng chữ cái. Vì định nghĩa này cũng sẽ giữ các thông báo hệ thống ở ngôn ngữ gốc nên nó chỉ nên được sử dụng trong phạm vi của tập lệnh.

So sánh bằng số có bộ tùy chọn kiểm tra riêng:

\$NUM1 -lt \$NUM2

Đánh giá xem NUM1 có nhỏ hơn NUM2 hay không.

\$NUM1 -gt \$NUM2

Đánh giá xem NUM1 có lớn hơn NUM2 hay không.

\$NUM1 -le \$NUM2

Đánh giá xem NUM1 nhỏ hơn hay bằng NUM2.

\$NUM1 -ge \$NUM2

Đánh giá xem NUM1 lớn hơn hay bằng NUM2.

\$NUM1 -eq \$NUM2

Đánh giá xem NUM1 có bằng NUM2 hay không.

\$NUM1 -ne \$NUM2

Đánh giá xem NUM1 có không bằng NUM2 hay không.

Tất cả các phiên kiểm tra đều có thể nhận được các hiệu chỉnh sau:

! EXPR

Đánh giá xem biểu thức EXPR có sai không.

EXPR1 -a EXPR2

Đánh giá xem cả EXPR1 và EXPR2 có đều đúng hay không.

EXPR1 -o EXPR2

Đánh giá xem ít nhất một trong hai biểu thức có đúng hay không.

Một cấu trúc điều kiện khác là `case` có thể được coi là một biến thể của cấu trúc `if`. Lệnh `case` sẽ thực thi một danh sách các lệnh đã cho nếu một mục được chỉ định — ví dụ như nội dung của một biến — có thể được tìm thấy trong danh sách các mục được phân tách bằng *các ống* (thanh dọc `|`) và được kết thúc bởi `)`. Tập lệnh mẫu sau đây cho thấy cách sử dụng cấu trúc `case` để biểu thị định dạng gói phần mềm tương ứng cho một bản phân phối Linux nhất định:

```
#!/bin/bash

DISTRO=$1
```

```

echo -n "Distribution $DISTRO uses "
case "$DISTRO" in
    debian | ubuntu | mint)
        echo -n "the DEB"
        ;;
    centos | fedora | opensuse )
        echo -n "the RPM"
        ;;
    *)
        echo -n "an unknown"
        ;;
esac
echo " package format."

```

Mỗi danh sách mẫu và lệnh liên quan phải được kết thúc bằng `;;`, `;&` hoặc `;;&`. Mẫu cuối cùng (dấu hoa thị) sẽ khớp nếu trước đó không có mẫu nào khác tương ứng. Lệnh `esac` (ngược lại của `case`) sẽ kết thúc cấu trúc `case`. Giả sử tệp lệnh mẫu trước đó được đặt tên là `script.sh` và nó được thực thi với `opensuse` làm đối số đầu tiên thì kết quả đầu ra sau sẽ được tạo:

```

$ ./script.sh opensuse
Distribution opensuse uses the RPM package format.

```

TIP

Bash có một tùy chọn gọi là `nocasematch` cho phép khớp mẫu không phân biệt chữ hoa chữ thường cho cấu trúc `case` và các lệnh điều kiện khác. Lệnh tích hợp sẵn `shopt` sẽ chuyển đổi các giá trị của cài đặt kiểm soát hành vi vỏ tùy chọn: `shopt -s` sẽ bật (*set*) tùy chọn đã cho và `shopt -u` sẽ tắt (*unset*) tùy chọn đã cho. Do đó, việc đặt `shopt -s nocasematch` trước cấu trúc `case` sẽ cho phép khớp mẫu không phân biệt chữ hoa chữ thường. Các tùy chọn được sửa đổi bởi `shopt` sẽ chỉ ảnh hưởng đến phiên hiện tại. Do đó, các tùy chọn được sửa đổi bên trong các tệp lệnh đang chạy trong một vỏ con — vốn là cách tiêu chuẩn để chạy tệp lệnh — sẽ không ảnh hưởng đến các tùy chọn của phiên chính.

Mục được tìm kiếm và các mẫu sẽ trải qua quá trình mở rộng dấu ngã, mở rộng tham số, thay thế lệnh và mở rộng số học. Nếu mục tìm kiếm được chỉ định bằng dấu trích dẫn, chúng sẽ bị xóa trước khi thử khớp.

Cấu trúc Vòng lặp

Các tệp lệnh thường được sử dụng như một công cụ để tự động hóa các tác vụ lặp đi lặp lại hoặc thực hiện cùng một bộ lệnh cho đến khi tiêu chí dừng được xác minh. Bash có ba lệnh vòng lặp — `for`, `until` và `while` — được thiết kế cho các cấu trúc vòng lặp khác biệt hơn một chút.

Cấu trúc `for` sẽ duyệt qua một danh sách các mục nhất định — thường là một danh sách các từ hoặc bất kỳ đoạn văn bản nào được phân tách bằng dấu cách — và thực hiện cùng một bộ lệnh trên mỗi mục đó. Trước mỗi lần lặp, lệnh `for` sẽ gán mục hiện tại cho một biến, sau đó các lệnh kèm theo có thể sử dụng mục này. Quá trình sẽ được lặp lại cho đến khi không còn mục nào nữa. Cú pháp của cấu trúc `for` là:

```
for VARNAME in LIST
do
    COMMANDS
done
```

`VARNAME` là tên biến vô tùy ý và `LIST` là bất kỳ một chuỗi thuật ngữ riêng biệt nào. Các ký tự phân cách hợp lệ chia tách các mục trong danh sách sẽ được xác định bởi biến môi trường `IFS`, theo mặc định là các ký tự *dấu cách*, *tab* và *dòng mới*. Danh sách các lệnh được thực thi sẽ được phân cách bằng lệnh `do` và `done`, vì vậy nên các lệnh có thể chiếm bao nhiêu dòng cũng được.

Trong ví dụ sau, lệnh `for` sẽ lấy từng mục từ danh sách được cung cấp — một chuỗi số — và gán nó cho biến `NUM`, mỗi lần gán một mục:

```
#!/bin/bash

for NUM in 1 1 2 3 5 8 13
do
    echo -n "$NUM is "
    if [ $(( $NUM % 2 )) -ne 0 ]
    then
        echo "odd."
    else
        echo "even."
    fi
done
```

Trong ví dụ này, một cấu trúc `if` lồng được sử dụng kết hợp với một biểu thức số học để đánh giá xem số trong biến `NUM` hiện tại là chẵn hay lẻ. Giả sử tệp lệnh mẫu trước đó có tên là `script.sh` và nó nằm trong thư mục hiện tại, kết quả đầu ra sau sẽ được tạo:

```
$ ./script.sh
1 is odd.
1 is odd.
2 is even.
3 is odd.
```

```
5 is odd.
8 is even.
13 is odd.
```

Bash cũng hỗ trợ một định dạng thay thế cho cấu trúc `for` với ký hiệu hai dấu ngoặc đơn. Ký hiệu này giống với cú pháp lệnh `for` trong ngôn ngữ lập trình C và nó đặc biệt hữu ích khi làm việc với mảng:

```
#!/bin/bash

SEQ=( 1 1 2 3 5 8 13 )

for (( IDX = 0; IDX < ${#SEQ[*]}; IDX++ ))
do
    echo -n "${SEQ[$IDX]} is "
    if [ $(( ${SEQ[$IDX]} % 2 )) -ne 0 ]
    then
        echo "odd."
    else
        echo "even."
    fi
done
```

Tập lệnh mẫu này sẽ tạo ra kết quả đầu ra giống hệt như ví dụ trước. Tuy nhiên, thay vì sử dụng biến `NUM` để lưu trữ từng mục một, biến `IDX` sẽ được sử dụng để theo dõi chỉ số mảng hiện tại theo thứ tự tăng dần, bắt đầu từ 0 và liên tục thêm vào khi nó ở dưới số lượng mục trong mảng `SEQ`. Mục thực tế sẽ được truy xuất từ vị trí mảng của nó với `${SEQ[$IDX]}`.

Theo cách tương tự, cấu trúc `until` sẽ thực thi một chuỗi lệnh cho đến khi một lệnh kiểm tra — giống như chính lệnh `test` — kết thúc với trạng thái 0 (thành công). Ví dụ: cấu trúc vòng lặp tương tự từ ví dụ trước có thể được triển khai với `until` như sau:

```
#!/bin/bash

SEQ=( 1 1 2 3 5 8 13 )

IDX=0

until [ $IDX -eq ${#SEQ[*]} ]
do
    echo -n "${SEQ[$IDX]} is "
    if [ $(( ${SEQ[$IDX]} % 2 )) -ne 0 ]
```

```

then
    echo "odd."
else
    echo "even."
fi
IDX=$(( $IDX + 1 ))
done

```

Cấu trúc `until` có thể yêu cầu nhiều lệnh hơn cấu trúc `for` nhưng nó lại có thể phù hợp hơn với tiêu chí dừng phi số được cung cấp bởi biểu thức `test` hoặc bất kỳ một lệnh nào khác. Quan trọng nhất là người dùng cần bao gồm các hành động đảm bảo tiêu chí dừng hợp lệ (chẳng hạn như mức tăng của biến đếm); nếu không, vòng lặp có thể sẽ chạy mãi mãi.

Lệnh `while` cũng tương tự như `until` nhưng nó sẽ tiếp tục lặp lại tập hợp lệnh nếu lệnh kiểm tra kết thúc với trạng thái 0 (thành công). Do đó, lệnh `until [$IDX -eq ${#SEQ[*]}]` từ ví dụ trước sẽ tương đương với `while [$IDX -lt ${#SEQ[*]}]` vì vòng lặp sẽ lặp lại trong khi chỉ số mảng sẽ *nhỏ hơn* tổng số mục trong mảng.

Một Ví dụ phức tạp hơn

Hãy tưởng tượng một người dùng muốn đồng bộ hóa định kỳ một tập hợp các tệp và thư mục của họ với một thiết bị lưu trữ khác được gắn tại một điểm gắn kết tùy ý trong hệ thống tệp và một hệ thống sao lưu đầy đủ tính năng thì lại quá dư thừa. Vì đây là hoạt động được thực hiện định kỳ nên tự động hóa bằng tập lệnh vỏ là một ứng dụng rất phù hợp cho trường hợp sử dụng này.

Tác vụ rất đơn giản: đồng bộ hóa mọi tệp và thư mục có trong danh sách, từ thư mục gốc được coi là đối số tệp lệnh đầu tiên đến thư mục đích được coi là đối số tệp lệnh thứ hai. Để giúp việc thêm hoặc xóa các mục khỏi danh sách trở nên dễ dàng hơn, nó sẽ được giữ trong một tệp riêng biệt là `~/ .sync .list`, một mục trên mỗi dòng:

```

$ cat ~/ .sync .list
Documents
To do
Work
Family Album
.config
.ssh
.bash_profile
.vimrc

```

Tệp này có chứa hỗn hợp các tệp và thư mục, một số còn có khoảng trống trong tên của chúng.

Đây là một kịch bản phù hợp cho lệnh Bash tích hợp sẵn `mapfile`. Lệnh này sẽ phân tích mọi nội dung văn bản đã cho, tạo một biến mảng từ đó và đặt mỗi dòng dưới dạng một mục mảng riêng lẻ. Tập lệnh sẽ có tên `sync.sh` và có chứa tập lệnh sau:

```
#!/bin/bash

set -ef

# List of items to sync
FILE=~/.sync.list

# Origin directory
FROM=$1

# Destination directory
TO=$2

# Check if both directories are valid
if [ ! -d "$FROM" -o ! -d "$TO" ]
then
    echo Usage:
    echo "$0 <SOURCEDIR> <DESTDIR>"
    exit 1
fi

# Create array from file
mapfile -t LIST < $FILE

# Sync items
for (( IDX = 0; IDX < ${#LIST[*]}; IDX++ ))
do
    echo -e "$FROM/${LIST[$IDX]} \u2192 $TO/${LIST[$IDX]}";
    rsync -qa --delete "$FROM/${LIST[$IDX]}" "$TO";
done
```

Hành động đầu tiên mà tập lệnh sẽ thực hiện là xác định lại hai tham số vỏ bằng lệnh `set`: tùy chọn `-e` sẽ thoát thực thi ngay lập tức nếu một lệnh thoát với trạng thái khác 0 và tùy chọn `-f` sẽ vô hiệu hóa tính năng khớp mẫu khối cho tên tập. Cả hai tùy chọn có thể được rút ngắn thành `-ef`. Đây không phải là bước bắt buộc nhưng nó sẽ giúp giảm thiểu khả năng xảy ra các hành vi không mong muốn.

Các chỉ dẫn định hướng ứng dụng thực tế của tập lệnh có thể được chia thành ba phần:

1. Thu thập và kiểm tra tham số tập lệnh

Biến `FILE` là đường dẫn đến tệp chứa danh sách các mục cần sao chép: `~/ .sync.list`. Các biến `FROM` và `TO` lần lượt là đường dẫn gốc và đích. Vì hai tham số cuối cùng này do người dùng cung cấp nên chúng sẽ trải qua một cuộc kiểm tra xác thực đơn giản được thực hiện bởi cấu trúc `if`: nếu bất kỳ tham số nào trong hai tham số này không phải là thư mục hợp lệ — được đánh giá bằng phiên kiểm tra `[! -d "$FROM" -o ! -d "$TO"]` — tập lệnh sẽ hiển thị một thông báo trợ giúp ngắn gọn và sau đó kết thúc với trạng thái thoát là 1.

2. Tải danh sách tệp và thư mục

Sau khi tất cả các tham số được xác định, một mảng chứa danh sách các mục cần sao chép sẽ được tạo bằng lệnh `mapfile -t LIST < $FILE`. Tùy chọn `-t` của `mapfile` sẽ xóa ký tự dòng mới ở cuối mỗi dòng trước khi đưa nó vào biến mảng có tên `LIST`. Nội dung của tệp được biểu thị bằng biến `FILE` — `~/ .sync.list` — được đọc thông qua chuyển hướng đầu vào.

3. Thực hiện sao chép và thông báo cho người dùng

Vòng lặp `for` sử dụng ký hiệu hai dấu ngoặc đơn để duyệt qua mảng của các mục với biến `IDX` theo dõi mức tăng chỉ số. Lệnh `echo` sẽ thông báo cho người dùng về từng mục được sao chép. Ký tự unicode đã thoát — `\u2192` — cho ký tự *mũi tên bên phải* sẽ hiện diện trong thông báo đầu ra, vì thế nên ta phải sử dụng tùy chọn `-e` của lệnh `echo`. Lệnh `rsync` sẽ chỉ sao chép có chọn lọc các phần tệp đã sửa đổi từ nguồn gốc, do đó ta nên sử dụng lệnh này cho các tác vụ như vậy. Các tùy chọn của `rsync` là `-q` và `-a` (có thể được gói gọn thành `-qa`) sẽ chặn các thông báo `rsync` và kích hoạt chế độ *lưu trữ* và tất cả các thuộc tính tệp sẽ được giữ nguyên. Tùy chọn `--delete` sẽ khiến `rsync` xóa một mục ở đích không còn tồn tại ở gốc nữa, vì vậy chúng ta nên sử dụng nó một cách cẩn thận.

Giả sử tất cả các mục trong danh sách đều tồn tại trong thư mục chính của người dùng `carol` (`/home/carol`) và thư mục đích `/media/carol/backup` trở đến một thiết bị lưu trữ ngoại vi được gắn kết, lệnh `sync.sh /home/carol /media/carol/backup` sẽ tạo ra kết quả đầu ra sau:

```
$ sync.sh /home/carol /media/carol/backup
/home/carol/Documents → /media/carol/backup/Documents
/home/carol/"To do" → /media/carol/backup/"To do"
/home/carol/Work → /media/carol/backup/Work
/home/carol/"Family Album" → /media/carol/backup/"Family Album"
/home/carol/.config → /media/carol/backup/.config
/home/carol/.ssh → /media/carol/backup/.ssh
/home/carol/.bash_profile → /media/carol/backup/.bash_profile
/home/carol/.vimrc → /media/carol/backup/.vimrc
```

Ví dụ này cũng giả định tệp lệnh được thực thi bởi siêu người dùng hoặc bởi người dùng `carol` vì hầu hết các tệp sẽ không thể đọc được bởi những người dùng khác. Nếu `script.sh` không nằm trong thư mục được liệt kê trong biến môi trường `PATH` thì nó phải được chỉ định bằng đường dẫn đầy đủ.

Bài tập Hướng dẫn

1. Làm cách nào để có thể sử dụng lệnh `test` để xác minh xem đường dẫn tệp được lưu trong biến `FROM` có mới hơn tệp có đường dẫn được lưu trong biến `TO` không?

2. Tệp lệnh sau sẽ in một dãy số từ 0 đến 9, nhưng thay vào đó nó in 0 vô thời hạn. Bạn cần phải làm gì để có được kết quả như mong đợi?

```
#!/bin/bash

COUNTER=0

while [ $COUNTER -lt 10 ]
do
    echo $COUNTER
done
```

3. Giả sử một người dùng đã viết một tệp lệnh yêu cầu danh sách tên người dùng được sắp xếp. Kết quả danh sách được trình bày như sau trên máy tính của họ:

```
carol
Dave
emma
Frank
Grace
henry
```

Tuy nhiên, danh sách này lại được sắp xếp như sau trên máy tính của đồng nghiệp của họ:

```
Dave
Frank
Grace
carol
emma
henry
```

Điều gì có thể giải thích được sự khác biệt này?

Bài tập Mở rộng

1. Làm thế nào để có thể sử dụng tất cả các đối số dòng lệnh của tập lệnh để khởi tạo một mảng Bash?

2. Tại sao lệnh `test 1 > 2` lại được đánh giá là đúng?

3. Làm cách nào để người dùng tạm thời thay đổi dấu phân cách trường mặc định thành ký tự dòng mới trong khi vẫn có thể hoàn nguyên nó về nội dung ban đầu?

Tóm tắt

Bài học này đã đi vào sâu hơn về các phiên kiểm tra có sẵn cho lệnh `test` cũng như các cấu trúc vòng lặp và điều kiện khác cần thiết để viết các tập lệnh vỏ phức tạp hơn. Một tập lệnh đồng bộ hóa tập đơn giản đã được đưa ra làm ví dụ về ứng dụng tập lệnh vỏ thực tế. Bài học đã đi qua các bước sau:

- Kiểm tra mở rộng cho cấu trúc điều kiện `if` và `case`.
- Cấu trúc vòng lặp Vỏ: `for`, `until` và `while`.
- Lặp qua mảng và tham số.

Các lệnh và quy trình đã được nhắc tới là:

`test`

Thực hiện so sánh giữa các mục được cung cấp cho lệnh.

`if`

Một cấu trúc logic được sử dụng trong các tập lệnh để đánh giá điều gì đó là đúng hay sai, sau đó thực thi lệnh nhánh dựa trên kết quả.

`case`

Đánh giá nhiều giá trị dựa trên một biến duy nhất. Việc thực thi lệnh tập lệnh sau đó sẽ được thực hiện tùy thuộc vào kết quả của lệnh `case`.

`for`

Lặp lại việc thực thi một lệnh dựa trên một tiêu chí nhất định.

`until`

Lặp lại việc thực thi một lệnh cho đến khi một biểu thức được đánh giá là sai.

`while`

Lặp lại việc thực thi một lệnh trong khi một biểu thức đã cho được đánh giá là đúng.

Đáp án Bài tập Hướng dẫn

1. Làm cách nào để có thể sử dụng lệnh `test` để xác minh xem đường dẫn tệp được lưu trong biến `FROM` có mới hơn tệp có đường dẫn được lưu trong biến `TO` không?

Lệnh `test "$FROM" -nt "$TO"` sẽ trả về mã trạng thái là 0 nếu tệp trong biến `FROM` mới hơn tệp trong biến `TO`.

2. Tệp lệnh sau sẽ in một dãy số từ 0 đến 9, nhưng thay vào đó nó in 0 vô thời hạn. Bạn cần phải làm gì để có được kết quả như mong đợi?

```
#!/bin/bash

COUNTER=0

while [ $COUNTER -lt 10 ]
do
    echo $COUNTER
done
```

Biến `COUNTER` phải được gia tăng. Điều này có thể được thực hiện bằng biểu thức số học `COUNTER=$(($COUNTER + 1))` để đạt được tiêu chí dừng và kết thúc vòng lặp.

3. Giả sử một người dùng đã viết một tệp lệnh yêu cầu danh sách tên người dùng được sắp xếp. Kết quả danh sách được trình bày như sau trên máy tính của họ:

```
carol
Dave
emma
Frank
Grace
henry
```

Tuy nhiên, danh sách này lại được sắp xếp như sau trên máy tính của đồng nghiệp của họ:

```
Dave
Frank
Grace
carol
emma
```

henry

Điều gì có thể giải thích được sự khác biệt này?

Việc sắp xếp dựa trên ngôn ngữ của hệ thống hiện tại. Để tránh sự mâu thuẫn, các tác vụ sắp xếp phải được thực hiện với biến môi trường LANG được đặt thành C.

Đáp án Bài tập Mở rộng

1. Làm thế nào để có thể sử dụng tất cả các đối số dòng lệnh của tập lệnh để khởi tạo một mảng Bash?

Các lệnh `PARAMS=($*)` hoặc `PARAMS=("$@")` sẽ tạo ra một mảng có tên `PARAMS` với tất cả các đối số.

2. Tại sao lệnh `test 1 > 2` lại được đánh giá là đúng?

Toán tử `>` được thiết kế để sử dụng với các phiên kiểm tra chuỗi chứ không phải các phiên kiểm tra số.

3. Làm cách nào để người dùng tạm thời thay đổi dấu phân cách trường mặc định thành ký tự dòng mới trong khi vẫn có thể hoàn nguyên nó về nội dung ban đầu?

Một bản sao của biến `IFS` có thể được lưu trữ trong một biến khác: `OLDIFS=$IFS`. Sau đó, dấu tách dòng mới sẽ được xác định bằng `IFS=$'\n'` và biến `IFS` có thể được hoàn nguyên trở lại bằng `IFS=$OLDIFS`.



Chủ đề 106: Giao diện Người dùng và Máy tính để bàn



106.1 Cài đặt và định cấu hình X11

Tham khảo các mục tiêu LPI

[LPIC-1 version 5.0, Exam 102, Objective 106.1](#)

Khối lượng

2

Các lĩnh vực kiến thức chính

- Hiểu về kiến trúc X11.
- Hiểu và nắm vững kiến thức cơ bản về tệp cấu hình X Window.
- Ghi đề các khía cạnh cụ thể của cấu hình Xorg (chẳng hạn như bố cục bàn phím).
- Hiểu về các thành phần của môi trường máy tính để bàn (chẳng hạn như trình quản lý hiển thị và trình quản lý cửa sổ).
- Quản lý quyền truy cập vào máy chủ X và hiển thị các ứng dụng trên máy chủ X từ xa.
- Hiểu về Wayland.

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- `/etc/X11/xorg.conf`
- `/etc/X11/xorg.conf.d/`
- `~/.xsession-errors`
- `xhost`
- `xauth`
- `DISPLAY`
- `X`



106.1 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	106 Giao diện Người dùng và Máy tính
Mục tiêu:	106.1 Cài đặt và định cấu hình X11
Bài:	1 trên 1

Giới thiệu

Hệ thống X Window là một "chồng" các phần mềm được sử dụng để hiển thị văn bản và đồ họa trên màn hình. Giao diện và thiết kế tổng thể của một máy khách X không do Hệ thống X Window quy định mà thay vào đó được xử lý bởi từng máy khách X riêng lẻ, một *trình quản lý cửa sổ* (ví dụ như Trình tạo cửa sổ Window Maker, Trình quản lý cửa sổ tab Tab Window Manager) hoặc một *môi trường máy tính hoàn chỉnh* (chẳng hạn như KDE, GNOME hoặc Xfce). Môi trường máy tính sẽ được đề cập tới trong bài học sau. Bài học này sẽ tập trung vào các kiến trúc cơ bản và các công cụ phổ biến cho Hệ thống X Window mà quản trị viên sẽ sử dụng để định cấu hình X.

Hệ thống X Window là một hệ thống đa nền tảng và chạy trên nhiều hệ điều hành khác nhau như Linux, BSD, Solaris và các hệ thống tương tự Unix khác. Ngoài ra, nó còn có các bản triển khai dành cho macOS của Apple và Microsoft Windows.

Phiên bản chính của giao thức X được sử dụng trong các bản phân phối Linux hiện đại là *X.org* phiên bản 11 (thường được viết là *X11*). Giao thức X là cơ chế giao tiếp giữa máy khách X và máy chủ X. Sự khác biệt giữa máy khách X và máy chủ X sẽ được thảo luận bên dưới đây.

NOTE

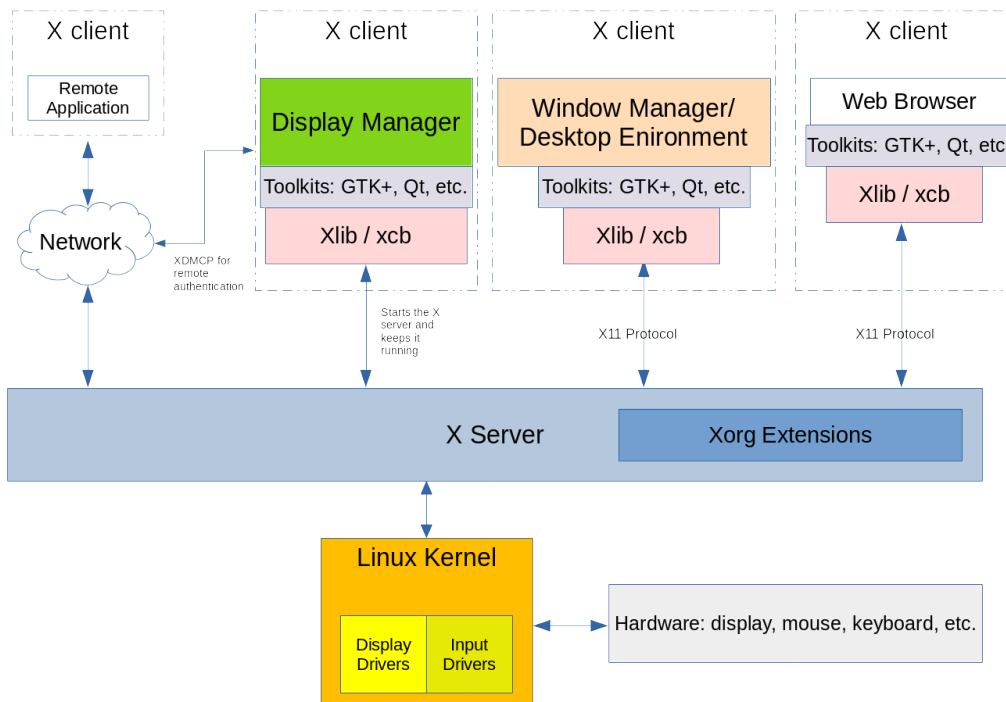
Tiền thân của Hệ thống X Window là một hệ thống cửa sổ có tên *W* và là nỗ lực

phát triển chung giữa IBM, DEC và MIT. Phần mềm này ra đời từ *Dự án Athena* vào năm 1984. Khi các nhà phát triển bắt đầu làm việc trên một máy chủ hiển thị mới, họ đã chọn chữ cái tiếp theo trong bảng chữ cái tiếng Anh là “X”. Sự phát triển của Hệ thống X Window hiện đang được kiểm soát bởi *Tập đoàn MIT X*.

Kiến trúc Hệ thống X Window

Hệ thống X Window cung cấp các cơ chế cho việc vẽ các hình dạng hai chiều cơ bản (và hình dạng ba chiều thông qua các tính năng mở rộng) trên một màn hình hiển thị. Nó được chia thành máy khách và máy chủ, và trong hầu hết các cài đặt cần có màn hình đồ họa thì cả hai thành phần này đều nằm trên cùng một máy tính. Các thành phần máy khách sẽ ở dưới dạng một ứng dụng (tương tự như các trình mô phỏng cửa sổ dòng lệnh, trò chơi hoặc trình duyệt web). Mỗi ứng dụng khách sẽ thông báo cho máy chủ X về vị trí và kích thước cửa sổ của nó trên màn hình máy tính. Máy khách cũng sẽ xử lý những gì được đưa vào cửa sổ đó và máy chủ X sẽ đưa bản vẽ được yêu cầu lên màn hình. Hệ thống X Window cũng xử lý đầu vào từ các thiết bị như chuột, bàn phím, bàn di chuột, v.v.

Cấu trúc cơ bản của một hệ thống X Window



Hệ thống X Window có khả năng kết nối mạng; nhiều máy khách X từ các máy tính khác nhau trên cùng một mạng có thể thực hiện các yêu cầu vẽ tới một máy chủ X từ xa. Lý do đằng sau điều

này là để quản trị viên hoặc người dùng có thể có quyền truy cập vào một ứng dụng đồ họa trên hệ thống từ xa mà hệ thống cục bộ của họ có thể không có.

Đặc điểm chính của Hệ thống X Window là nó mang tính mô-đun. Trong suốt quá trình phát triển của Hệ thống X Window, các tính năng mới hơn đã được phát triển và bổ sung vào khuôn khổ của nó. Các thành phần mới này chỉ được thêm dưới dạng tiện ích mở rộng cho máy chủ X nên giao thức X11 cốt lõi vẫn luôn được giữ nguyên vẹn. Các tiện ích mở rộng này được chứa trong các tệp thư viện *Xorg*. Ví dụ về các thư viện Xorg bao gồm `libXrandr`, `libXcursor`, `libX11`, `libxkbfile` cũng như một số thư viện khác; mỗi thư viện đều sẽ cung cấp chức năng mở rộng cho máy chủ X.

Trình quản lý hiển thị sẽ cung cấp thông tin đăng nhập đồ họa cho hệ thống. Hệ thống này có thể là máy tính cục bộ hoặc một máy tính ở trong mạng. Trình quản lý hiển thị sẽ được khởi chạy sau khi máy tính khởi động và sẽ bắt đầu phiên máy chủ X cho người dùng đã được xác thực. Trình quản lý hiển thị cũng chịu trách nhiệm duy trì hoạt động của máy chủ X. GDM, SDDM và LightDM là các ví dụ về Trình quản lý hiển thị.

Mỗi phiên bản của máy chủ X đang chạy đều sẽ có một *tên hiển thị* để nhận dạng nó. Tên hiển thị sẽ có nội dung như sau:

```
hostname:displaynumber.screennumber
```

Tên hiển thị cũng sẽ hướng dẫn ứng dụng đồ họa về nơi nó được kết xuất và kết xuất trên máy chủ nào (nếu sử dụng một kết nối X từ xa).

`hostname` (tên máy chủ) là tên của hệ thống sẽ hiển thị ứng dụng. Nếu tên máy chủ bị thiếu trong tên hiển thị thì máy chủ cục bộ sẽ được coi là máy chủ.

`displaynumber` (số màn hình hiển thị) tham chiếu đến lượng “màn hình” đang được sử dụng dù đó là một màn hình máy tính xách tay hay nhiều màn hình trên một máy trạm. Mỗi phiên máy chủ X đang chạy đều sẽ được cấp một số hiển thị bắt đầu từ 0.

`screennumber` (số màn hình vật lý) mặc định sẽ là 0. Nó sẽ là 0 nếu chỉ một màn hình vật lý hoặc nhiều màn hình vật lý được cấu hình để hoạt động như một màn hình. Khi tất cả các màn hình trong thiết lập đa màn hình hiển thị được kết hợp thành một màn hình logic, các cửa sổ ứng dụng có thể được di chuyển tự do giữa các màn hình. Trong trường hợp tất cả các màn hình đều được định cấu hình để hoạt động độc lập với nhau, mỗi màn hình sẽ chứa các cửa sổ ứng dụng mở bên trong chúng và không thể di chuyển các cửa sổ này từ màn hình này sang màn hình khác. Mỗi màn hình độc lập sẽ được gán một số riêng của nó. Nếu chỉ sử dụng một màn hình logic thì dấu chấm và số màn hình sẽ bị bỏ qua.

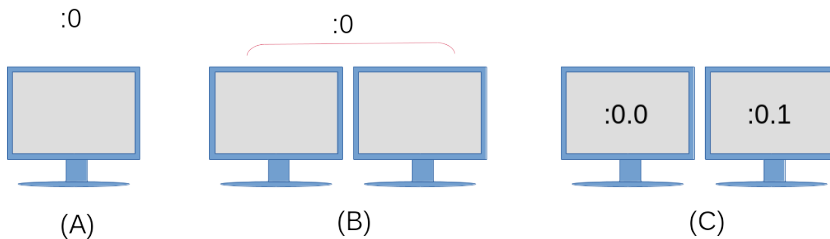
Tên hiển thị của một phiên X đang chạy sẽ được lưu trữ trong biến môi trường `DISPLAY`:

```
$ echo $DISPLAY
:0
```

Chi tiết của đầu ra sẽ như sau:

1. Máy chủ X đang được sử dụng nằm trên hệ thống cục bộ, do đó mà không có gì được in ở bên trái dấu hai chấm.
2. Phiên máy chủ X hiện tại là phiên đầu tiên như được biểu thị bằng số 0 ngay sau dấu hai chấm.
3. Chỉ có một màn hình logic được sử dụng nên số màn hình không được hiển thị.

Để minh họa rõ hơn những khái niệm này, hãy tham khảo sơ đồ sau: Ví dụ về Cấu hình hiển thị



(A)

Một màn hình hiển thị đơn với một cấu hình hiển thị duy nhất và chỉ một màn hình.

(B)

Được định cấu hình dưới dạng một màn hình, với hai màn hình vật lý được định cấu hình thành một màn hình. Cửa sổ ứng dụng có thể được di chuyển tự do giữa hai màn hình.

(C)

Một cấu hình hiển thị duy nhất (như được biểu thị bằng `:0`); tuy nhiên, mỗi màn hình hiển thị lại là một màn hình độc lập. Cả hai màn hình vẫn sẽ dùng chung các thiết bị đầu vào như bàn phím và chuột; tuy nhiên, một ứng dụng được mở trên màn hình `:0.0` sẽ không thể chuyển sang màn hình `:0.1` và ngược lại.

Để khởi động một ứng dụng trên một màn hình cụ thể, hãy gán số màn hình cho biến môi trường `DISPLAY` trước khi khởi chạy ứng dụng:

```
$ DISPLAY=:0.1 firefox &
```

Lệnh này sẽ khởi động trình duyệt web Firefox trên màn hình bên phải trong sơ đồ trên. Một số bộ công cụ cũng sẽ cung cấp các tùy chọn dòng lệnh để hướng dẫn ứng dụng chạy trên một màn

hình cụ thể. Hãy xem ví dụ về `--screen` và `--display` trong trang hướng dẫn `gtk-options(7)`.

Cấu hình Máy chủ X

Thông thường, tệp cấu hình chính được sử dụng để định cấu hình máy chủ X là tệp `/etc/X11/xorg.conf`. Trên các bản phân phối Linux hiện đại, máy chủ X sẽ tự cấu hình trong thời gian chạy khi máy chủ X khởi động và do đó không có tệp `xorg.conf` nào tồn tại.

Tệp `xorg.conf` được chia thành các khối riêng biệt được gọi là *các phần* (sections). Mỗi phần đều sẽ bắt đầu bằng thuật ngữ `Section` và theo sau là *tên phần* để nói đến cấu hình thành phần. Mỗi `Section` sẽ được kết thúc bằng một `EndSection` tương ứng. Tệp `xorg.conf` điển hình sẽ chứa các phần sau:

InputDevice

Được sử dụng để định cấu hình một mẫu bàn phím hoặc chuột cụ thể.

InputClass

Trong các bản phân phối Linux hiện đại, phần này thường được tìm thấy trong một tệp cấu hình riêng biệt nằm trong `/etc/X11/xorg.conf.d/`. `InputClass` được sử dụng để định cấu hình một *hạng* (class) thiết bị phần cứng như bàn phím và chuột thay vì một phần cứng cụ thể. Dưới đây là một ví dụ về tệp `/etc/X11/xorg.conf.d/00-keyboard.conf`:

```
Section "InputClass"
    Identifier "system-keyboard"
    MatchIsKeyboard "on"
    Option "XkbLayout" "us"
    Option "XkbModel" "pc105"
EndSection
```

Tùy chọn cho `XkbLayout` sẽ xác định bố cục của các phím trên bàn phím (chẳng hạn như Dvorak, thuận tay trái hoặc tay phải, QWERTY và ngôn ngữ). Tùy chọn cho `XkbModel` sẽ được sử dụng để xác định loại bàn phím đang sử dụng. Ta có thể tìm thấy một bảng có chứa mẫu, bố cục và mô tả của chúng trong `xkeyboard-config(7)`. Các tệp được liên kết với bố cục bàn phím có thể được tìm thấy trong `/usr/share/X11/xkb`. Ví dụ về bố cục bàn phím Polytonic tiếng Hy Lạp trên máy tính Chromebook sẽ giống như sau:

```
Section "InputClass"
    Identifier "system-keyboard"
    MatchIsKeyboard "on"
    Option "XkbLayout" "gr(polytonic)"
```

```
Option "XkbModel" "chromebook"
EndSection
```

Ngoài ra, bố cục của bàn phím có thể được sửa đổi trong phiên X đang chạy bằng lệnh `setxkbmap`. Dưới đây là ví dụ về lệnh thiết lập bố cục Polytonic Hy Lạp trên máy tính Chromebook:

```
$ setxkbmap -model chromebook -layout "gr(polytonic)"
```

Cài đặt này sẽ chỉ tồn tại trong thời gian sử dụng phiên X. Để thực hiện những thay đổi đó vĩnh viễn, hãy sửa đổi tệp `/etc/X11/xorg.conf.d/00-keyboard.conf` và thêm các cài đặt cần thiết.

NOTE

Lệnh `setxkbmap` sử dụng *Tiện ích mở rộng bàn phím X (XKB)*. Đây là một ví dụ về chức năng bổ sung của Hệ thống X Window thông qua việc sử dụng các tiện ích mở rộng.

Các bản phân phối Linux hiện đại có cung cấp lệnh `localectl` thông qua `systemd`. Lệnh này cũng có thể được sử dụng để sửa đổi bố cục bàn phím và sẽ tự động tạo tệp cấu hình `/etc/X11/xorg.conf.d/00-keyboard.conf`. Một lần nữa, đây là ví dụ về thiết lập bàn phím Polytonic tiếng Hy Lạp trên Chromebook, lần này là sử dụng lệnh `localectl`:

```
$ localectl --no-convert set-x11-keymap "gr(polytonic)" chromebook
```

Tùy chọn `--no-convert` được sử dụng ở đây để ngăn `localectl` sửa đổi sơ đồ bàn phím bảng điều khiển của máy chủ.

Monitor

Phần `Monitor` mô tả màn hình vật lý được sử dụng và nơi nó được kết nối. Dưới đây là một ví dụ về cấu hình hiển thị màn hình phần cứng được kết nối với cổng hiển thị thứ hai và được sử dụng làm màn hình chính.

```
Section "Monitor"
    Identifier "DP2"
    Option "Primary" "true"
EndSection
```

Device

Phần `Device` mô tả thẻ màn hình (video card) vật lý được sử dụng. Phần này cũng sẽ chứa mô-

đơn hạt nhân được sử dụng làm trình điều khiển cho thẻ màn hình cùng với vị trí vật lý của nó trên bo mạch chủ.

```
Section "Device"
    Identifier "Device0"
    Driver     "i915"
    BusID      "PCI:0:2:0"
EndSection
```

Screen

Phần Screen liên kết các phần Monitor và Device với nhau. Một ví dụ về phần Screen có thể trông giống như sau:

```
Section "Screen"
    Identifier "Screen0"
    Device     "Device0"
    Monitor    "DP2"
EndSection
```

ServerLayout

Phần ServerLayout sẽ nhóm tất cả các phần như chuột, bàn phím và màn hình vào một giao diện Hệ thống X Window.

```
Section "ServerLayout"
    Identifier "Layout-1"
    Screen     "Screen0" 0 0
    InputDevice "mouse1" "CorePointer"
    InputDevice "system-keyboard" "CoreKeyboard"
EndSection
```

NOTE

Không phải tất cả các phần đều có thể được tìm thấy trong tệp cấu hình. Trong trường hợp thiếu một phần, các giá trị mặc định sẽ được cung cấp bởi phiên bản máy chủ X đang chạy.

Các tệp cấu hình do người dùng chỉ định cũng nằm trong `/etc/X11/xorg.conf.d/`. Các tệp cấu hình do bản phân phối cung cấp nằm ở `/usr/share/X11/xorg.conf.d/`. Các tệp cấu hình nằm trong `/etc/X11/xorg.conf.d/` sẽ được phân tích cú pháp trước tệp `/etc/X11/xorg.conf` nếu nó tồn tại trên hệ thống.

Lệnh `xdpynfo` được sử dụng trên máy tính để hiển thị thông tin về phiên bản máy chủ X đang

chạy. Dưới đây là đầu ra mẫu từ lệnh:

```
$ xdpinfo
name of display:      :0
version number:      11.0
vendor string:       The X.Org Foundation
vendor release number: 12004000
X.Org version:       1.20.4
maximum request size: 16777212 bytes
motion buffer size:  256
bitmap unit, bit order, padding:  32, LSBFirst, 32
image byte order:    LSBFirst
number of supported pixmap formats:  7
supported pixmap formats:
    depth 1, bits_per_pixel 1, scanline_pad 32
    depth 4, bits_per_pixel 8, scanline_pad 32
    depth 8, bits_per_pixel 8, scanline_pad 32
    depth 15, bits_per_pixel 16, scanline_pad 32
    depth 16, bits_per_pixel 16, scanline_pad 32
    depth 24, bits_per_pixel 32, scanline_pad 32
    depth 32, bits_per_pixel 32, scanline_pad 32
keycode range:      minimum 8, maximum 255
focus: None
number of extensions:  25
    BIG-REQUESTS
    Composite
    DAMAGE
    DOUBLE-BUFFER
    DRI3
    GLX
    Generic Event Extension
    MIT-SCREEN-SAVER
    MIT-SHM
    Present
    RANDR
    RECORD
    RENDER
    SECURITY
    SHAPE
    SYNC
    X-Resource
    XC-MISC
    XFIXES
    XFree86-VidModeExtension
```

```

XINERAMA
XInputExtension
XKEYBOARD
XTEST
XVideo
default screen number:    0
number of screens:       1

screen #0:
  dimensions:    3840x1080 pixels (1016x286 millimeters)
  resolution:    96x96 dots per inch
  depths (7):    24, 1, 4, 8, 15, 16, 32
  root window id:  0x39e
  depth of root window:  24 planes
  number of colormaps:  minimum 1, maximum 1
  default colormap:  0x25
  default number of colormap cells:  256
  preallocated pixels:  black 0, white 16777215
  options:    backing-store WHEN MAPPED, save-unders NO
  largest cursor:  3840x1080
  current input event mask:  0xda0033
    KeyPressMask           KeyReleaseMask           EnterWindowMask
    LeaveWindowMask        StructureNotifyMask       SubstructureNotifyMask
    SubstructureRedirectMask PropertyChangeMask       ColormapChangeMask
  number of visuals:  270
  ...

```

Các phần đầu ra quan trọng hơn sẽ được in đậm, chẳng hạn như tên của màn hình hiển thị (giống với nội dung của biến môi trường DISPLAY), thông tin phiên bản của máy chủ X đang sử dụng, số lượng và danh sách các tiện ích mở rộng Xorg đang được sử dụng và các thông tin khác về bản thân màn hình.

Tạo Tập cấu hình Xorg cơ bản

Mặc dù X sẽ tạo cấu hình của nó sau khi khởi động hệ thống trên các bản cài đặt Linux hiện đại nhưng tệp `xorg.conf` vẫn có thể được sử dụng. Để tạo tệp `/etc/X11/xorg.conf` vĩnh viễn, hãy chạy lệnh sau:

```
$ sudo Xorg -configure
```

NOTE

Nếu đã có một phiên X đang chạy, ta sẽ cần chỉ định một DISPLAY khác trong lệnh của mình. Ví dụ:

```
$ sudo Xorg :1 -configure
```

Trên một số bản phân phối Linux, lệnh X có thể được sử dụng thay cho Xorg vì X là một liên kết tượng trưng đến Xorg.

Một tệp `xorg.conf.new` sẽ được tạo trong thư mục làm việc hiện tại của người dùng. Nội dung của tệp này được lấy từ những gì máy chủ X đã tìm thấy có sẵn trong phần cứng và trình điều khiển trên hệ thống cục bộ. Để sử dụng tệp này, chúng ta cần di chuyển nó tới thư mục `/etc/X11/` và đổi tên thành `xorg.conf`:

```
$ sudo mv xorg.conf.new /etc/X11/xorg.conf
```

NOTE

Các trang hướng dẫn sau đây sẽ cung cấp thêm thông tin về các thành phần của Hệ thống X Window: `xorg.conf(5)`, `Xserver(1)`, `X(1)` và `Xorg(1)`.

Wayland

Wayland là một giao thức hiển thị mới hơn được thiết kế để thay thế Hệ thống X Window. Nhiều bản phân phối Linux hiện đại đang sử dụng nó làm máy chủ hiển thị mặc định. Bản chất mục đích của Wayland là để sử dụng ít tài nguyên hệ thống hơn và có dung lượng cài đặt nhỏ hơn X. Dự án này được bắt đầu vào năm 2010 và vẫn đang trong quá trình phát triển tích cực bao gồm cả những nỗ lực của các nhà phát triển X.org từ quá khứ tới hiện tại.

Không giống như Hệ thống X Window, không có phiên bản máy chủ nào chạy giữa máy khách và hạt nhân. Thay vào đó, cửa sổ máy khách hoạt động với mã riêng của nó hoặc mã của bộ công cụ (chẳng hạn như Gtk+ hoặc Qt) để cung cấp kết xuất. Để thực hiện kết xuất, một yêu cầu sẽ được gửi tới nhân Linux thông qua giao thức Wayland. Hạt nhân sẽ chuyển tiếp yêu cầu thông qua giao thức Wayland tới *trình tạo ảnh* Wayland được sử dụng để xử lý đầu vào thiết bị, quản lý cửa sổ và bố cục. Trình tạo ảnh là một phần của hệ thống được sử dụng để kết hợp các phần tử được kết xuất thành đầu ra hình ảnh trên màn hình.

Hầu hết các bộ công cụ hiện đại như Gtk+ 3 và Qt 5 đều đã được cập nhật để cho phép hiển thị trên Hệ thống X Window hoặc một máy tính chạy Wayland. Hiện tại không phải ứng dụng độc lập nào cũng đều được viết để hỗ trợ hiển thị trong Wayland. Đối với các ứng dụng và khuôn khổ vẫn đang hướng tới chạy Hệ thống X Window, ứng dụng có thể chạy bên trong *XWayland*. Hệ thống XWayland là một máy chủ X riêng biệt chạy trong một máy khách Wayland và do đó sẽ hiển thị nội dung của cửa sổ máy khách trong một phiên bản máy chủ X độc lập.

Giống như Hệ thống X Window sử dụng biến môi trường `DISPLAY` để theo dõi các màn hình đang được sử dụng, giao thức Wayland sử dụng biến môi trường `WAYLAND_DISPLAY`. Dưới đây là đầu ra

mẫu từ một hệ thống chạy giao thức hiển thị Wayland:

```
$ echo $WAYLAND_DISPLAY  
wayland-0
```

Biến môi trường này sẽ không có sẵn trên các hệ thống chạy X.

Bài tập Hướng dẫn

1. Bạn sẽ sử dụng lệnh nào để xác định tiện ích mở rộng Xorg nào có sẵn trên hệ thống?

2. Bạn vừa nhận được một con chuột 10 nút hoàn toàn mới cho máy tính của mình. Tuy nhiên, nó sẽ yêu cầu cấu hình bổ sung để tất cả các nút có thể hoạt động được bình thường. Nếu không sửa đổi phần còn lại của cấu hình máy chủ X, bạn sẽ sử dụng thư mục nào để tạo tệp cấu hình mới cho con chuột này và phần cấu hình cụ thể nào sẽ được sử dụng trong tệp này?

3. Thành phần nào của bản cài đặt Linux sẽ chịu trách nhiệm duy trì hoạt động của máy chủ X?

4. Khoá chuyển dòng lệnh nào được sử dụng với lệnh X để tạo một tệp cấu hình `xorg.conf` mới?

Bài tập Mở rộng

1. Giả sử rằng biến môi trường DISPLAY đang được xem trong trình mô phỏng cửa sổ dòng lệnh trên một màn hình độc lập thứ ba, nội dung của biến môi trường DISPLAY sẽ là gì trên hệ thống có tên lab01 sử dụng một cấu hình hiển thị duy nhất?

2. Lệnh nào có thể được sử dụng để tạo tệp cấu hình bàn phím để Hệ thống X Window sử dụng?

3. Trên một bản cài đặt Linux thông thường, người dùng có thể chuyển sang cửa sổ dòng lệnh ảo bằng cách nhấn các phím `Ctrl` + `Alt` + `F1-F6` trên bàn phím. Bạn đã được yêu cầu thiết lập hệ thống kiosk có giao diện đồ họa và cần tắt tính năng này để ngăn chặn việc giả mạo hệ thống trái phép. Bạn quyết định tạo tệp cấu hình `/etc/X11/xorg.conf.d/10-kiosk.conf`. Bằng cách sử dụng phần `ServerFlags` (được sử dụng để đặt các tùy chọn Xorg chung trên máy chủ), ta sẽ cần chỉ định tùy chọn nào? Hãy xem lại trang hướng dẫn `xorg(1)` để tìm tùy chọn.

Tóm tắt

Bài học này đề cập đến Hệ thống X Window khi nó được sử dụng trên Linux. Hệ thống X Window được sử dụng để vẽ hình ảnh và văn bản trên màn hình khi chúng được xác định trong các tệp cấu hình. Hệ thống X Window thường được sử dụng để định cấu hình các thiết bị đầu vào như chuột và bàn phím. Bài học này thảo luận các ý chính sau:

- Kiến trúc Hệ thống X Window ở mức cao.
- Tệp cấu hình nào được sử dụng để định cấu hình Hệ thống X Window và vị trí của chúng trên hệ thống tệp.
- Cách sử dụng biến môi trường `DISPLAY` trên hệ thống chạy X.
- Giới thiệu ngắn gọn về giao thức hiển thị Wayland.

Các lệnh và tệp cấu hình đã được nhắc tới là:

- Sửa đổi bố cục bàn phím trong bản cài đặt Xorg với `setxkbmap` và `localectl`.
- Lệnh Xorg để tạo tệp cấu hình `/etc/X11/xorg.conf` mới.
- Nội dung của tệp cấu hình Xorg có tại: `/etc/X11/xorg.conf`, `/etc/X11/xorg.conf.d/` và `/usr/share/X11/xorg.conf.d/`.
- Lệnh `xdpypinfo` để hiển thị thông tin chung về phiên máy chủ X đang chạy.

Đáp án Bài tập Hướng dẫn

1. Bạn sẽ sử dụng lệnh nào để xác định tiện ích mở rộng Xorg nào có sẵn trên hệ thống?

```
$ xdpinfo
```

2. Bạn vừa nhận được một con chuột 10 nút hoàn toàn mới cho máy tính của mình. Tuy nhiên, nó sẽ yêu cầu cấu hình bổ sung để tất cả các nút có thể hoạt động được bình thường. Nếu không sửa đổi phần còn lại của cấu hình máy chủ X, bạn sẽ sử dụng thư mục nào để tạo tệp cấu hình mới cho con chuột này và phần cấu hình cụ thể nào sẽ được sử dụng trong tệp này?

Các cấu hình do người dùng xác định phải được đặt trong `/etc/X11/xorg.conf.d/` và phần cụ thể cần thiết cho cấu hình chuột này sẽ là `InputDevice`.

3. Thành phần nào của bản cài đặt Linux sẽ chịu trách nhiệm duy trì hoạt động của máy chủ X?

Trình quản lý hiển thị.

4. Khoá chuyển dòng lệnh nào được sử dụng với lệnh `X` để tạo một tệp cấu hình `xorg.conf` mới?

```
-configure
```

Hãy nhớ rằng lệnh `X` là một liên kết tượng trưng đến lệnh `Xorg`.

Đáp án Bài tập Mở rộng

- Giả sử rằng biến môi trường DISPLAY đang được xem trong trình mô phỏng cửa sổ dòng lệnh trên một màn hình độc lập thứ ba, nội dung của biến môi trường DISPLAY sẽ là gì trên hệ thống có tên lab01 sử dụng một cấu hình hiển thị duy nhất?

```
$ echo $DISPLAY
lab01:0.2
```

- Lệnh nào có thể được sử dụng để tạo tệp cấu hình bàn phím để Hệ thống X Window sử dụng?

```
$ localectl
```

- Trên một bản cài đặt Linux thông thường, người dùng có thể chuyển sang cửa sổ dòng lệnh ảo bằng cách nhấn các phím `Ctrl` + `Alt` + `F1-F6` trên bàn phím. Bạn đã được yêu cầu thiết lập hệ thống kiosk có giao diện đồ họa và cần tắt tính năng này để ngăn chặn việc giả mạo hệ thống trái phép. Bạn quyết định tạo tệp cấu hình `/etc/X11/xorg.conf.d/10-kiosk.conf`. Bằng cách sử dụng phần `ServerFlags` (được sử dụng để đặt các tùy chọn Xorg chung trên máy chủ), ta sẽ cần chỉ định tùy chọn nào? Hãy xem lại trang hướng dẫn `xorg(1)` để tìm tùy chọn.

```
Section "ServerFlags"
    Option "DontVTSwitch" "True"
EndSection
```



106.2 Máy tính để bàn đồ họa

Tham khảo các mục tiêu LPI

[LPIC-1 version 5.0, Exam 102, Objective 106.2](#)

Khối lượng

1

Các lĩnh vực kiến thức chính

- Hiểu về các môi trường máy tính để bàn chính
- Hiểu về các giao thức được sử dụng để truy cập các phiên máy tính để bàn từ xa

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- KDE
- Gnome
- Xfce
- X11
- XDMCP
- VNC
- Spice
- RDP



106.2 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	106 Giao diện Người dùng và Máy tính
Mục tiêu:	106.2 Máy tính đồ họa
Bài:	1 trên 1

Giới thiệu

Các hệ điều hành dựa trên Linux được biết đến với giao diện dòng lệnh tiên tiến nhưng cũng có thể sẽ gây khó dễ cho những người dùng không rành về kỹ thuật. Với ý định làm cho việc sử dụng máy tính trở nên trực quan hơn, sự kết hợp giữa màn hình độ phân giải cao với các thiết bị trở đã tạo ra giao diện người dùng hướng tới hình ảnh. Mặc dù giao diện dòng lệnh yêu cầu người dùng phải có kiến thức về tên chương trình và các tùy chọn cấu hình của chúng, với *giao diện người dùng đồ họa* (GUI), chức năng chương trình có thể được kích hoạt bằng cách trở đến các phần tử hình ảnh quen thuộc giúp cho quá trình nghiên cứu trở nên đỡ vất vả hơn. Hơn nữa, giao diện người dùng đồ họa là giao diện phù hợp nhất cho đa phương tiện và các hoạt động trực quan khác.

Thật vậy, giao diện đồ họa người dùng gần như đồng nghĩa với giao diện máy tính và hầu hết các bản phân phối Linux đều được cài đặt giao diện đồ họa theo mặc định. Tuy nhiên, không có một chương trình nguyên khối nào có sẵn trong hệ thống Linux chịu trách nhiệm cho các máy tính đồ họa đầy đủ tính năng. Thay vào đó, mỗi máy tính đồ họa trên thực tế là một tập hợp lớn các chương trình và phần phụ thuộc của chúng được thay đổi tùy theo từng bản phân phối hoặc sở thích cá nhân của người dùng.

Hệ thống X Window

Trong Linux và các hệ điều hành tương tự Unix khác mà nó được sử dụng, *Hệ thống X Window* (hay còn được gọi là *X11* hoặc chỉ *X*) sẽ cung cấp các tài nguyên cấp thấp liên quan đến kết xuất giao diện đồ họa và tương tác của người dùng với nó, chẳng hạn như:

- Xử lý các sự kiện đầu vào như chuyển động của chuột hoặc tổ hợp phím.
- Khả năng cắt, sao chép và dán nội dung văn bản giữa các ứng dụng tách biệt.
- Giao diện lập trình các chương trình khác sử dụng các phần tử đồ họa để vẽ.

Mặc dù Hệ thống X Window chịu trách nhiệm điều khiển màn hình đồ họa hiển thị (bản thân trình điều khiển video là một phần của X) nhưng nó không nhằm tới mục đích tự mình vẽ các phần tử hình ảnh phức tạp. Hình dạng, màu sắc, sắc thái và bất kỳ hiệu ứng hình ảnh nào khác đều được tạo ra bởi ứng dụng chạy trên X. Cách tiếp cận này mang lại cho ứng dụng nhiều không gian để tạo giao diện tùy chỉnh nhưng nó cũng có thể dẫn đến tiêu hao phát triển vượt quá phạm vi ứng dụng và sự không nhất quán về hình thức và hoạt động khi so sánh với các giao diện chương trình khác.

Từ góc nhìn của nhà phát triển, việc giới thiệu *môi trường máy tính* tạo điều kiện thuận lợi cho việc lập trình GUI gắn liền với việc phát triển ứng dụng cơ bản. Trong khi đó, trên phương diện của người dùng, nó sẽ mang lại một trải nghiệm nhất quán giữa các ứng dụng riêng biệt. Môi trường máy tính tập hợp các giao diện lập trình, thư viện và các chương trình hỗ trợ hợp tác để cung cấp các khái niệm thiết kế truyền thống vẫn đang không ngừng phát triển.

Môi trường Máy tính

GUI của máy tính truyền thống bao gồm nhiều cửa sổ khác nhau — thuật ngữ *cửa sổ* được sử dụng ở đây để chỉ bất kỳ một khu vực màn hình độc lập nào — được liên kết với các tiến trình đang chạy. Vì Hệ thống X Window chỉ cung cấp các tính năng tương tác cơ bản nên trải nghiệm người dùng đầy đủ sẽ phụ thuộc vào các thành phần do môi trường máy tính cung cấp.

Thành phần quan trọng nhất của môi trường máy tính - *Trình quản lý Cửa sổ* - chịu trách nhiệm kiểm soát vị trí và trang trí cửa sổ. Trình quản lý cửa sổ sẽ thêm thanh tiêu đề vào cửa sổ, các nút điều khiển - thường được liên kết với các hành động thu nhỏ, phóng to và đóng - và quản lý việc chuyển đổi giữa các cửa sổ đang mở.

NOTE

Các khái niệm cơ bản trong giao diện đồ họa của máy tính đều xuất phát từ những ý tưởng được lấy từ không gian văn phòng làm việc thực tế. Nói một cách ví von, màn hình máy tính là màn hình nền nơi đặt các đối tượng như tài liệu và thư mục. Một cửa sổ ứng dụng có nội dung của một tài liệu sẽ bắt chước các thao tác vật lý

như điền vào biểu mẫu hoặc vẽ một bức tranh. Tương tự như bàn làm việc thực tế, máy tính cũng có các phụ kiện phần mềm như sổ ghi chú, đồng hồ, lịch, v.v.; hầu hết chúng đều là các bản sao của các phụ kiện “thực”.

Tất cả các môi trường máy tính đều cung cấp một trình quản lý cửa sổ phù hợp với giao diện của *bộ công cụ tiện ích* của nó. Bộ công cụ tiện ích là các phần tử trực quan mang tính thông tin hoặc tương tác như các nút hoặc trường nhập văn bản được phân phối bên trong cửa sổ ứng dụng. Các thành phần máy tính tiêu chuẩn — như trình khởi chạy ứng dụng, thanh tác vụ, v.v. — và bản thân trình quản lý cửa sổ đều phụ thuộc vào các bộ công cụ tiện ích đó để cấu thành giao diện của mình.

Các thư viện phần mềm (như *GTK+* và *Qt*) cung cấp các tiện ích mà lập trình viên có thể sử dụng để xây dựng các giao diện đồ họa phức tạp cho ứng dụng của họ. Từ trước tới nay, các ứng dụng được phát triển bằng *GTK+* trông sẽ không giống với các ứng dụng được tạo bằng *Qt* và ngược lại, nhưng sự phát triển của các tính năng hỗ trợ chủ đề của môi trường máy tính ngày nay đã khiến cho sự khác biệt này ngày càng được thu hẹp lại.

Nhìn chung, *GTK+* và *Qt* đều cung cấp các tính năng giống nhau về tiện ích. Các phần tử tương tác đơn giản có thể sẽ tương tự như nhau và khó có thể phân biệt được, trong khi các tiện ích tổng hợp - chẳng hạn như cửa sổ hộp thoại được ứng dụng sử dụng để mở hoặc lưu tệp - lại có thể trông khá khác biệt. Tuy nhiên, các ứng dụng được xây dựng bằng các bộ công cụ riêng biệt có thể chạy song song với nhau bất kể bộ công cụ tiện ích được các thành phần màn hình khác sử dụng là gì.

Ngoài các thành phần máy tính cơ bản có thể được coi là các chương trình riêng lẻ, môi trường máy tính cũng theo đuổi hình ảnh ẩn dụ của một chiếc bàn làm việc thực thụ bằng cách cung cấp một bộ ứng dụng phụ kiện tối thiểu được phát triển theo cùng một nguyên tắc thiết kế. Các biến thể của các ứng dụng sau thường được cung cấp bởi tất cả các môi trường máy tính phổ biến:

Các ứng dụng liên quan đến hệ thống

Trình mô phỏng cửa sổ dòng lệnh, trình quản lý tệp, trình quản lý cài đặt gói, công cụ cấu hình hệ thống.

Giao tiếp và Internet

Trình quản lý danh bạ, ứng dụng email khách, trình duyệt web.

Ứng dụng văn phòng

Lịch, máy tính, trình soạn thảo văn bản.

Môi trường máy tính có thể bao gồm nhiều dịch vụ và ứng dụng khác: trình màn hình chào đăng nhập, trình quản lý phiên, trình giao tiếp giữa các tiến trình, trình tạo khóa, v.v. Chúng cũng kết hợp các tính năng do dịch vụ hệ thống được bên thứ ba cung cấp (như *PulseAudio* cho âm thanh

và CUPS để in). Các tính năng này không cần môi trường đồ họa để hoạt động nhưng môi trường máy tính vẫn cung cấp giao diện người dùng đồ họa để tạo điều kiện thuận lợi cho việc cấu hình và vận hành các tài nguyên đó.

Các Môi trường Máy tính phổ biến

Nhiều hệ điều hành độc quyền chỉ hỗ trợ một môi trường máy tính chính thức duy nhất gắn liền với bản phát hành cụ thể của chúng và không thay đổi được. Ngược lại, các hệ điều hành dựa trên Linux sẽ hỗ trợ các tùy chọn môi trường máy tính khác nhau có thể được sử dụng cùng với X. Mỗi môi trường máy tính đều có các tính năng riêng nhưng chúng thường có một số ý tưởng thiết kế chung:

- Trình khởi chạy ứng dụng liệt kê các ứng dụng tích hợp sẵn và của bên thứ ba có sẵn trong hệ thống.
- Quy tắc xác định các ứng dụng mặc định liên quan đến loại tệp và giao thức.
- Công cụ cấu hình để tùy chỉnh giao diện và hoạt động của môi trường máy tính.

Gnome là một trong những môi trường máy tính phổ biến nhất và cũng là lựa chọn đầu tiên trong các bản phân phối như Fedora, Debian, Ubuntu, SUSE Linux Enterprise, Red Hat Enterprise Linux, CentOS, v.v. Trong phiên bản 3, Gnome đã mang đến những thay đổi lớn về giao diện và cấu trúc của nó để dần tách khỏi phép ẩn dụ về bàn làm việc và giới thiệu *Vỏ Gnome* làm giao diện mới.



Figure 1. Hoạt động của Vỏ Gnome

Trình khởi chạy toàn màn hình cho mục đích chung *Hoạt động Vỏ Gnome* đã thay thế trình khởi chạy ứng dụng và thanh tác vụ truyền thống. Tuy nhiên, người dùng vẫn có thể sử dụng Gnome 3

với giao diện cũ bằng cách chọn tùy chọn *Gnome Classic* trong màn hình đăng nhập.

KDE là một hệ sinh thái ứng dụng và nền tảng phát triển rất rộng. Phiên bản môi trường máy tính mới nhất của nó, *KDE Plasma*, được sử dụng theo mặc định trong openSUSE, Mageia, Kubuntu, v.v. Việc áp dụng thư viện Qt là tính năng nổi bật của KDE và đã mang lại cho nó vẻ ngoài không thể nhầm lẫn được cùng với rất nhiều ứng dụng gốc. KDE thậm chí còn cung cấp một công cụ cấu hình để đảm bảo sự gắn kết trực quan với các ứng dụng GTK+.

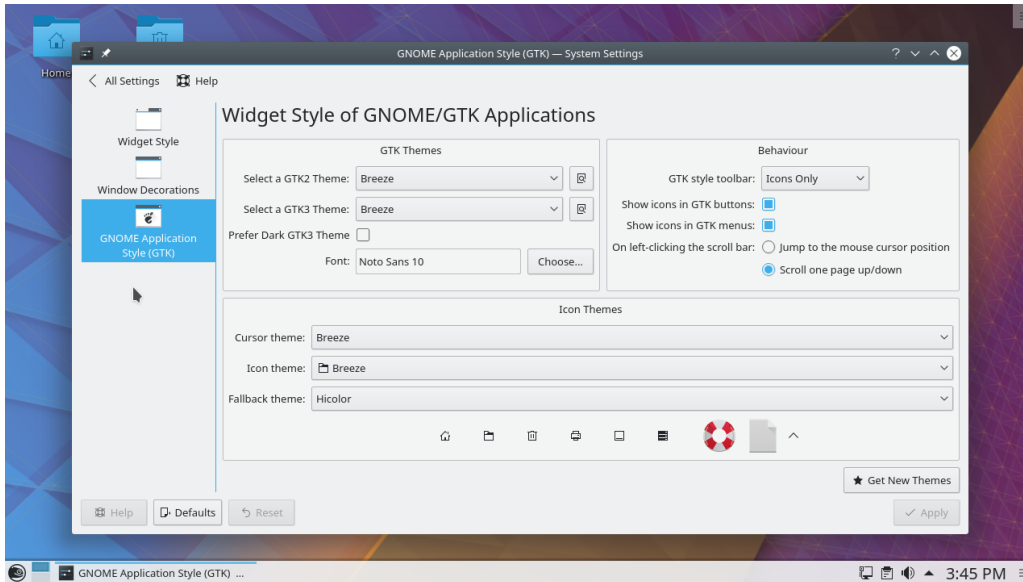


Figure 2. Cài đặt KDE GTK

Xfce là môi trường máy tính hướng tới tính thẩm mỹ nhưng không tiêu tốn nhiều tài nguyên máy. Cấu trúc của nó được mô đun hóa cao để cho phép người dùng kích hoạt và hủy kích hoạt các thành phần theo nhu cầu và sở thích của họ.

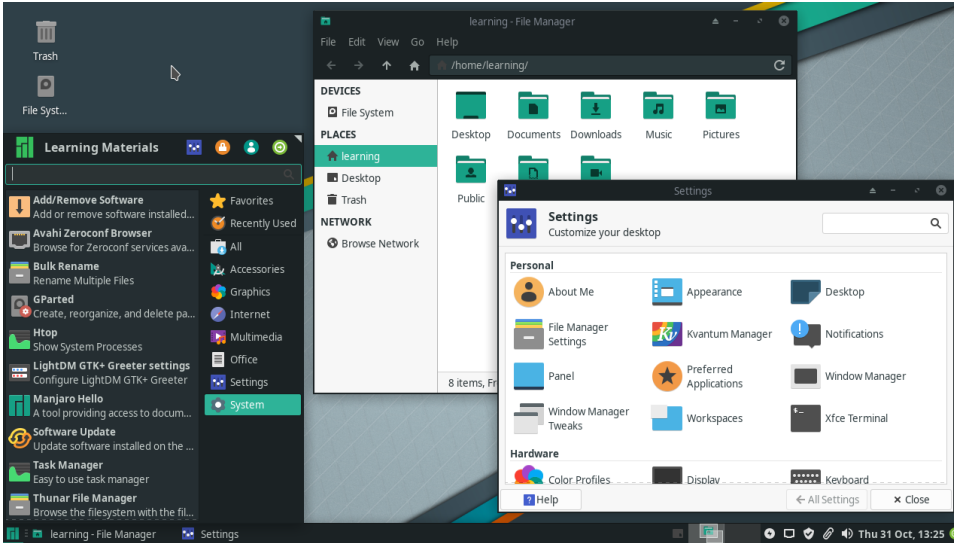


Figure 3. Máy tính Xfce

Có nhiều môi trường máy tính khác dành cho Linux thường được cung cấp bởi các vòng phân phối thay thế. Ví dụ: bản phân phối Linux Mint cung cấp hai môi trường máy tính gốc: *Cinnamon* (một nhánh của Gnome 3) và *MATE* (một nhánh của Gnome 2). *LXDE* là môi trường máy tính được thiết kế để tiêu thụ ít tài nguyên; điều này đã khiến nó trở thành lựa chọn tốt để cài đặt trên các thiết bị cũ hoặc máy tính bo mạch đơn. Mặc dù không cung cấp tất cả các tính năng của các môi trường máy tính nặng nhưng *LXDE* vẫn cung cấp tất cả các tính năng cơ bản được mong đợi từ giao diện người dùng đồ họa hiện đại.

TIP

Phím tắt đóng vai trò quan trọng trong việc tương tác với môi trường máy tính. Một số phím tắt — chẳng hạn như **Alt + Tab** để chuyển giữa các cửa sổ hoặc **Ctrl + C** để sao chép văn bản — có thể phổ biến trên tất cả các môi trường máy tính, nhưng mỗi môi trường máy tính cũng vẫn sẽ có các phím tắt riêng. Các phím tắt sẽ được tìm thấy trong công cụ cấu hình bàn phím nơi người dùng có thể thêm hoặc sửa đổi các phím tắt do môi trường máy tính cung cấp.

Khả năng tương tác trên Máy tính

Sự đa dạng của môi trường máy tính trong các hệ điều hành dựa trên Linux đã đặt ra một thách thức: làm thế nào để chúng hoạt động chính xác với các ứng dụng đồ họa hoặc dịch vụ hệ thống của bên thứ ba mà không cần phải triển khai hỗ trợ cụ thể cho từng môi trường đó. Các phương pháp và thông số kỹ thuật được chia sẻ trên các môi trường máy tính đã cải thiện đáng kể trải nghiệm người dùng và giải quyết nhiều vấn đề về phát triển. Lý do là bởi các ứng dụng đồ họa sẽ phải tương tác với môi trường máy tính hiện tại bất kể môi trường máy tính mà chúng được thiết kế ban đầu là gì. Ngoài ra, quan trọng nhất là người dùng phải giữ nguyên cài đặt chung của máy tính nếu về sau họ quyết định thay đổi lựa chọn môi trường máy tính của mình.

Tổ chức *freedesktop.org* duy trì một lượng lớn thông số kỹ thuật về khả năng tương tác trên máy tính. Việc áp dụng đầy đủ các đặc tính không phải là bắt buộc nhưng rất nhiều trong số chúng được sử dụng rộng rãi:

Vị trí thư mục

Nơi chứa các cài đặt cá nhân và các tệp dành riêng cho người dùng khác.

Các mục trên máy tính

Các ứng dụng dòng lệnh có thể chạy trong môi trường máy tính thông qua bất kỳ trình mô phỏng cửa sổ dòng lệnh nào, nhưng nếu cung cấp tất cả các ứng dụng đó trong quá trình khởi chạy ứng dụng thì sẽ rất dễ gây nhầm lẫn. Các mục trên máy tính là các tệp văn bản kết thúc bằng `.desktop` được môi trường máy tính sử dụng để thu thập thông tin về các ứng dụng có sẵn và cách để sử dụng chúng.

Khởi động ứng dụng tự động

Các mục trên màn hình cho biết ứng dụng sẽ tự khởi động sau khi người dùng đăng nhập.

Kéo và thả

Cách ứng dụng xử lý các sự kiện kéo và thả.

Thùng rác

Vị trí chung của các tệp bị trình quản lý tệp xóa cũng như các phương pháp lưu trữ và xóa tệp khỏi đó.

Chủ đề biểu tượng

Định dạng chung cho các thư viện biểu tượng có thể hoán đổi.

Tính dễ sử dụng do môi trường máy tính mang lại có một nhược điểm so với các giao diện văn bản như vỏ là khả năng cung cấp quyền truy cập từ xa. Mặc dù người dùng có thể dễ dàng truy cập môi trường vỏ dòng lệnh của máy từ xa bằng các công cụ như `ssh` nhưng việc truy cập từ xa vào môi trường đồ họa sẽ yêu cầu các phương pháp khác và có thể sẽ không đạt được hiệu suất thỏa đáng trên các kết nối chậm.

Truy cập phi cục bộ

Hệ thống X Window áp dụng thiết kế dựa trên *các màn hình hiển thị* độc lập mà trong đó, cùng một *trình quản lý hiển thị X* có thể kiểm soát nhiều phiên máy tính đồ họa cùng một lúc. Về bản chất, một màn hình hiển thị cũng tương tự như một cửa sổ dòng lệnh văn bản: cả hai đều đề cập đến một máy hoặc ứng dụng phần mềm được sử dụng làm điểm nhập để thiết lập một phiên hệ điều hành độc lập. Mặc dù thiết lập phổ biến nhất bao gồm một phiên đồ họa đơn lẻ chạy trên

máy cục bộ nhưng người dùng cũng có thể thực hiện được các thiết lập khác không thông dụng:

- Chuyển đổi giữa các phiên máy tính đồ họa đang hoạt động trong cùng một máy.
- Nhiều hơn một bộ thiết bị hiển thị (ví dụ: màn hình, bàn phím, chuột) được kết nối với cùng một máy, mỗi thiết bị sẽ điều khiển phiên máy tính đồ họa của riêng mình.
- Phiên máy tính đồ họa từ xa, trong đó, giao diện đồ họa sẽ được gửi qua mạng tới một màn hình hiển thị từ xa.

Phiên kết nối từ xa được hỗ trợ cục bộ bởi X sử dụng *Giao thức điều khiển trình quản lý hiển thị X* (XDMCP) để giao tiếp với màn hình hiển thị từ xa. Do mức sử dụng băng thông cao, XDMCP hiếm khi được sử dụng qua internet hoặc trong mạng LAN tốc độ thấp. Các vấn đề bảo mật cũng là mối lo ngại đối với XDMCP: màn hình hiển thị cục bộ sẽ giao tiếp với một trình quản lý hiển thị X từ xa đặc quyền để thực thi các quy trình từ xa. Do đó, một lỗ hổng có thể khiến nó có thể thực thi các lệnh đặc quyền tùy ý trên máy từ xa.

Hơn nữa, XDMCP sẽ yêu cầu cả hai đầu kết nối sử dụng các phiên của X. Điều này có thể khiến cho tác vụ trở nên không khả thi nếu Hệ thống X Windows không có sẵn trên tất cả các máy liên quan. Trong thực tế, các phương pháp khác hiệu quả và ít xâm lấn hơn sẽ được sử dụng để thiết lập các phiên làm việc đồ họa từ xa.

Điện toán mạng ảo (VNC) là một công cụ độc lập với nền tảng để xem và kiểm soát các môi trường máy tính từ xa bằng giao thức *Bộ đệm Khung từ xa* (RFB). Thông qua nó, các sự kiện do bàn phím và chuột cục bộ tạo ra sẽ được truyền đến kết nối từ xa, từ đó sẽ gửi lại mọi cập nhật màn hình để được hiển thị cục bộ. Nhiều máy chủ VNC có thể chạy trên cùng một máy, nhưng mỗi máy chủ VNC sẽ cần một cổng TCP riêng trong giao diện mạng chấp nhận các yêu cầu phiên đến. Theo quy ước, máy chủ VNC đầu tiên nên sử dụng cổng TCP 5900, máy chủ thứ hai nên sử dụng cổng 5901, v.v.

Máy chủ VNC không cần đặc quyền để chạy. Ví dụ: một người dùng thông thường có thể đăng nhập vào tài khoản từ xa của họ và từ đó khởi động máy chủ VNC của riêng họ. Sau đó, trong máy cục bộ, bất kỳ ứng dụng khách VNC nào cũng có thể được sử dụng để truy cập kết nối từ xa (giả sử có thể truy cập được các cổng mạng tương ứng). Tệp `~/ .vnc/xstartup` là một tệp lệnh vỏ được máy chủ VNC thực thi khi nó khởi động và có thể được sử dụng để xác định môi trường máy tính nào mà máy chủ VNC sẽ cung cấp cho máy khách VNC. Điều quan trọng cần lưu ý là VNC sẽ không cung cấp các phương thức xác thực và mã hóa hiện đại. Do đó, nó nên được sử dụng cùng với ứng dụng của bên thứ ba có cung cấp các tính năng trên. Các phương pháp liên quan đến đường hầm VPN và SSH thường được sử dụng để bảo mật các kết nối VNC.

Giao thức máy tính từ xa (RDP) chủ yếu được sử dụng để truy cập từ xa vào máy tính của hệ điều hành *Microsoft Windows* thông qua cổng mạng TCP 3389. Mặc dù sử dụng giao thức RDP độc quyền của Microsoft nhưng triển khai máy khách được sử dụng trong hệ thống Linux là các

chương trình mã nguồn mở được cấp phép theo *Giấy phép Công cộng GNU (GPL)* và không có hạn chế pháp lý nào về việc sử dụng.

Giao thức đơn giản cho môi trường điện toán độc lập (Spice) bao gồm một bộ công cụ nhằm truy cập vào môi trường máy tính của các hệ thống ảo hóa, có thể là trong máy cục bộ hoặc ở một địa điểm từ xa. Ngoài ra, giao thức Spice còn cung cấp các tính năng gốc để tích hợp hệ thống cục bộ và từ xa như khả năng truy cập các thiết bị cục bộ (ví dụ như loa âm thanh và thiết bị USB được kết nối) từ máy từ xa và chia sẻ tệp giữa hai hệ thống.

Chúng ta có các lệnh máy khách cụ thể để kết nối với từng giao thức kết nối từ xa này, nhưng máy khách kết nối từ xa *Remmina* có cung cấp một giao diện đồ họa tích hợp tạo điều kiện thuận lợi cho quá trình kết nối và có thể tùy chọn lưu trữ cài đặt kết nối để sử dụng sau này. *Remmina* có các tiện ích bổ sung cho từng giao thức riêng lẻ và cho XDMCP, VNC, RDP và Spice. Việc lựa chọn công cụ phù hợp phụ thuộc vào hệ điều hành, chất lượng kết nối mạng và những tính năng có sẵn của môi trường kết nối từ xa.

Bài tập Hướng dẫn

1. Loại ứng dụng nào cung cấp các phiên vỏ có cửa sổ trong môi trường máy tính?

2. Do sự đa dạng của môi trường máy tính Linux, cùng một ứng dụng có thể có nhiều phiên bản, mỗi phiên bản sẽ phù hợp cho một bộ công cụ tiện ích nhất định. Ví dụ: ứng dụng khách bittorrent *Transmission* có hai phiên bản: *transmission-gtk* và *transmission-qt*. Phiên bản nào có thể đảm bảo tích hợp tối đa với KDE?

3. Môi trường máy tính Linux nào được khuyên dùng cho các máy tính bo mạch đơn giá rẻ và có ít khả năng xử lý?

Bài tập Mở rộng

1. Có hai cách để sao chép và dán văn bản trong Hệ thống X Window: sử dụng tổ hợp phím `Ctrl + C` và `Ctrl + V` truyền thống (cũng có sẵn trong menu cửa sổ) hoặc nhấp nút ở giữa chuột để dán văn bản hiện được chọn. Cách thích hợp để sao chép và dán văn bản từ trình mô phỏng cửa sổ dòng lệnh là gì?

2. Hầu hết các môi trường máy tính đều gán phím tắt `Alt + F2` cho cửa sổ *Chạy chương trình* (Run program) nơi các chương trình có thể được thực thi theo kiểu dòng lệnh. Trong KDE, lệnh nào sẽ thực thi trình mô phỏng cửa sổ dòng lệnh mặc định?

3. Giao thức nào sẽ phù hợp nhất để truy cập một kết nối Windows từ xa từ môi trường máy tính Linux?

Tóm tắt

Bài học này cung cấp một cái nhìn tổng quan về máy tính đồ họa có sẵn cho hệ thống Linux. Riêng Hệ thống X Window chỉ cung cấp các tính năng giao diện đơn giản, do đó mà môi trường máy tính sẽ mở rộng trải nghiệm của người dùng trong giao diện cửa sổ đồ họa. Bài học đã đi qua các chủ đề sau:

- Giao diện đồ họa và các khái niệm về Hệ thống X Window.
- Môi trường máy tính có sẵn cho Linux.
- Điểm tương đồng và khác biệt giữa các môi trường máy tính.
- Cách truy cập một môi trường máy tính từ xa.

Các khái niệm và chương trình được đề cập tới là:

- Hệ thống X Window.
- Môi trường máy tính phổ biến: KDE, Gnome, Xfce.
- Các giao thức truy cập từ xa: XDMCP, VNC, RDP, Spice.

Đáp án Bài tập Hướng dẫn

1. Loại ứng dụng nào cung cấp các phiên vỏ có cửa sổ trong môi trường máy tính?

Bất kỳ trình mô phỏng cửa sổ dòng lệnh nào như Konsole, Gnome terminal, xterm, v.v. cũng sẽ cấp quyền truy cập vào một phiên vỏ tương tác cục bộ.

2. Do sự đa dạng của môi trường máy tính Linux, cùng một ứng dụng có thể có nhiều phiên bản, mỗi phiên bản sẽ phù hợp cho một bộ công cụ tiện ích nhất định. Ví dụ: ứng dụng khách bittorrent *Transmission* có hai phiên bản: *transmission-gtk* và *transmission-qt*. Phiên bản nào có thể đảm bảo tích hợp tối đa với KDE?

KDE được xây dựng dựa trên thư viện Qt, vì vậy phiên bản Qt — *transmission-qt* — nên được cài đặt.

3. Môi trường máy tính Linux nào được khuyên dùng cho các máy tính bo mạch đơn giá rẻ và có ít khả năng xử lý?

Các môi trường máy tính cơ bản không sử dụng quá nhiều hiệu ứng hình ảnh (chẳng hạn như Xfce và LXDE).

Đáp án Bài tập Mở rộng

1. Có hai cách để sao chép và dán văn bản trong Hệ thống X Window: sử dụng tổ hợp phím `Ctrl` + `C` và `Ctrl` + `V` truyền thống (cũng có sẵn trong menu cửa sổ) hoặc nhấp nút ở giữa chuột để dán văn bản hiện được chọn. Cách thích hợp để sao chép và dán văn bản từ trình mô phỏng cửa sổ dòng lệnh là gì?

Các phiên vỏ tương tác chỉ định tổ hợp phím `Ctrl` + `C` để dừng thực thi chương trình; do đó, người dùng nên sử dụng nút giữa.

2. Hầu hết các môi trường máy tính đều gán phím tắt `Alt` + `F2` cho cửa sổ *Chạy chương trình* (Run program) nơi các chương trình có thể được thực thi theo kiểu dòng lệnh. Trong KDE, lệnh nào sẽ thực thi trình mô phỏng cửa sổ dòng lệnh mặc định?

Lệnh `konsole` sẽ thực thi trình mô phỏng cửa sổ dòng lệnh của KDE nhưng các thuật ngữ chung như *terminal* cũng có thể sử dụng được.

3. Giao thức nào sẽ phù hợp nhất để truy cập một kết nối Windows từ xa từ môi trường máy tính Linux?

Giao thức kết nối từ xa (RDP) vì nó được hỗ trợ nguyên bản bởi cả Windows và Linux.



106.3 Trợ năng

Tham khảo các mục tiêu LPI

LPIC-1 version 5.0, Exam 102, Objective 106.3

Khối lượng

1

Các lĩnh vực kiến thức chính

- Kiến thức cơ bản về cài đặt hình ảnh và chủ đề.
- Kiến thức cơ bản về công nghệ trợ năng.

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- Chủ đề máy tính để bàn có độ tương phản cao/chữ in lớn.
- Trình đọc màn hình.
- Màn hình chữ nổi.
- Kính lúp màn hình.
- Bàn phím trên màn hình.
- Phím dính/ Phím lập.
- Phím Chậm/Nảy/Chuyển đổi.
- Phím chuột.
- Cử chỉ.
- Nhận diện giọng nói.



106.3 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	106 Giao diện Người dùng và Máy tính
Mục tiêu:	106.3 Trợ năng
Bài:	1 trên 1

Giới thiệu

Môi trường máy tính Linux có nhiều cài đặt và công cụ để điều chỉnh giao diện cho người dùng có hạn chế về khả năng sử dụng. Các thiết bị giao diện thông thường của con người (như màn hình, bàn phím và chuột/bàn di chuột) có thể được cấu hình lại cho từng trường hợp để khắc phục tình trạng suy giảm thị lực hoặc suy giảm khả năng chuyển động.

Ví dụ: người dùng có thể điều chỉnh bảng màu của màn hình để phục vụ tốt hơn cho người dùng bị mù màu. Ngoài ra, những người bị chấn thương lặp đi lặp lại cũng có thể tận dụng các phương pháp gõ và trở thay thế.

Một số tính năng trợ năng này được cung cấp bởi chính môi trường máy tính (như Gnome hoặc KDE), còn các tính năng khác sẽ được cung cấp bởi các chương trình bổ sung. Trong trường hợp của các tính năng bổ sung, quan trọng nhất là người dùng cần chọn được công cụ tích hợp tốt nhất với môi trường máy tính của họ để đạt được chất lượng hỗ trợ tốt.

Cài đặt Trợ năng

Tất cả các bản phân phối Linux chính đều cung cấp các tính năng trợ năng tương tự như nhau và

có thể được tùy chỉnh bằng mô-đun cấu hình có trong trình quản lý cài đặt đi kèm với môi trường máy tính. Mô-đun cài đặt trợ năng trong Gnome được gọi là *Trợ năng phổ quát* (Universal Access) và trong KDE là *Cài đặt hệ thống* (System Settings), *Cá nhân hóa* (Personalization) và *Trợ năng* (Accessibility). Các môi trường máy tính khác (như *Xfce*) cũng gọi nó là *Trợ năng* trong trình quản lý cài đặt đồ họa của chúng. Tuy nhiên, chúng sẽ cung cấp ít bộ chức năng hơn so với Gnome và KDE.

Ví dụ: Gnome có thể được định cấu hình để hiển thị vĩnh viễn menu Trợ năng phổ quát ở góc trên bên phải màn hình nơi người dùng có thể nhanh chóng bật các tùy chọn trợ năng. Ví dụ: nó có thể được sử dụng để thay thế cảnh báo âm thanh bằng cảnh báo trực quan, nhờ đó mà người dùng khiếm thính sẽ nhận biết được cảnh báo hệ thống một cách dễ dàng hơn. Mặc dù KDE không có menu truy cập nhanh tương tự nhưng tính năng cảnh báo trực quan cũng có sẵn và được gọi là *chuông trực quan*.

Hỗ trợ Bàn phím và Chuột

Hoạt động của bàn phím và chuột mặc định có thể được sửa đổi để giải quyết các khó khăn nhất định về vấn đề di chuyển. Các tổ hợp phím, tốc độ lặp lại phím tự động và việc nhấn phím ngoài ý muốn có thể là những trở ngại đáng kể đối với người dùng bị suy giảm khả năng chuyển động cánh tay. Những thiếu sót khi gõ này được giải quyết bằng ba tính năng trợ năng liên quan đến bàn phím: *Phím dính* (Sticky keys), *Phím nảy* (Bounce keys) và *Phím chậm* (Slow keys).

Tính năng *Phím dính* có thể được tìm thấy ở phần *Hỗ trợ đánh máy* (Typing Assist) trong phần cấu hình Trợ năng phổ quát của Gnome. Nó cho phép người dùng nhập các phím tắt, mỗi lần một phím. Khi được bật, các tổ hợp phím như **Ctrl** + **C** không cần phải được nhấn và giữ cùng lúc. Trước tiên, người dùng có thể nhấn phím **Ctrl**, sau đó thả phím rồi nhấn phím **C**. Trong KDE, tùy chọn này nằm trong tab *Phím sửa đổi* (Modifier Keys) trong cửa sổ cài đặt Trợ năng. KDE cũng cung cấp tùy chọn *Phím khóa* (Locking Keys): nếu được bật, các phím **Alt**, **Ctrl** và **Shift** sẽ ở trạng thái "bị nhấn giữ" nếu người dùng nhấn chúng hai lần, tương tự như hành vi của phím Caps Lock. Giống như tính năng Caps Lock, người dùng sẽ cần nhấn lại phím tương ứng để trở về trạng thái ban đầu.

Tính năng *Phím nảy* sẽ cố gắng ngăn chặn những lần nhấn phím ngoài ý muốn bằng cách đặt độ trễ, nghĩa là một lần nhấn phím mới sẽ chỉ được chấp nhận sau khi một khoảng thời gian xác định đã trôi qua kể từ lần nhấn phím cuối cùng. Người dùng bị run tay có thể thấy tính năng *Phím nảy* rất hữu ích trong việc tránh nhấn một phím nhiều lần khi họ chỉ định nhấn một lần. Trong Gnome, tính năng này chỉ liên quan đến việc lặp lại một phím giống nhau; trong khi đó, ở KDE nó còn liên quan đến bất kỳ thao tác nhấn phím nào khác và được tìm thấy trong tab *Bộ lọc bàn phím* (Keyboard Filters).

Tính năng *Phím chậm* cũng sẽ giúp tránh đi việc gõ phím vô tình. Khi được bật, *Phím chậm* sẽ yêu cầu người dùng giữ phím trong một khoảng thời gian nhất định trước khi được chấp nhận. Tùy

thuộc vào nhu cầu của người dùng, việc điều chỉnh độ lặp lại tự động trong khi giữ một phím cũng có thể sẽ hữu ích. Tính năng này có sẵn trong mục cài đặt bàn phím.

Người dùng có thể bật và tắt các tính năng trợ năng của Phím dính và Phím chậm bằng *Cử chỉ kích hoạt* (Activation Gestures) được thực hiện trên bàn phím. Trong KDE, người dùng nên chọn tùy chọn *Sử dụng cử chỉ để kích hoạt phím dính và phím chậm* (Use gestures for activating sticky keys and slow keys) để bật *Cử chỉ kích hoạt*. Trong khi ở Gnome, tính năng này được gọi là *Bật bằng bàn phím* (Enable by Keyboard) trong cửa sổ cấu hình *Hỗ trợ đánh máy*. Khi *Cử chỉ kích hoạt* được bật, tính năng Phím dính sẽ được kích hoạt sau khi nhấn phím Shift năm lần liên tiếp. Để kích hoạt tính năng Phím chậm, phím Shift phải được giữ trong 8 giây liên tiếp.

Người dùng thích sử dụng bàn phím thay vì chuột hoặc bàn di chuột có thể sử dụng phím tắt để di chuyển trong môi trường máy tính. Hơn nữa, một tính năng được gọi là *Phím chuột* sẽ cho phép người dùng tự điều khiển con trỏ chuột bằng bàn phím số. Tính năng này có trong bàn phím máy tính để bàn và máy tính xách tay cỡ lớn.

Bàn phím số được sắp xếp thành một lưới vuông nên mỗi số sẽ tương ứng với một hướng: phím **2** di chuyển con trỏ xuống dưới, **4** di chuyển con trỏ sang trái, **7** di chuyển con trỏ về phía bắc- tây, v.v... Theo mặc định, số **5** sẽ tương ứng với việc nhấp chuột trái.

Trong khi ở Gnome chỉ có một công tắc để bật tùy chọn Phím chuột trong cửa sổ cài đặt Trợ năng phổ quát thì trong KDE, cài đặt Phím chuột được đặt tại *Cài đặt hệ thống, Chuột, Điều hướng bàn phím* (Keyboard Navigation) và các tùy chọn như tốc độ và khả năng tăng tốc cũng có thể được tùy chỉnh.

TIP

Phím chậm, Phím dính, Phím nảy và Phím chuột là các tính năng trợ năng được cung cấp bởi AccessX - một tài nguyên trong phần mở rộng bàn phím X của Hệ thống X Window. Cài đặt AccessX cũng có thể được sửa đổi từ dòng lệnh bằng lệnh `xkbset`.

Chuột hoặc bàn di chuột có thể được sử dụng để tạo đầu vào bàn phím khi việc sử dụng bàn phím là quá khó khăn hoặc không thể thực hiện được. Nếu công tắc *Bàn phím màn hình* (Screen Keyboard) trong cài đặt Trợ năng Phổ quát của Gnome được bật thì bàn phím ảo sẽ xuất hiện mỗi khi con trỏ ở trong trường văn bản và văn bản mới sẽ được nhập bằng cách nhấp vào các phím bằng chuột hoặc màn hình cảm ứng giống như bàn phím ảo của điện thoại thông minh.

KDE và các môi trường máy tính khác có thể sẽ không cung cấp bàn phím màn hình theo mặc định, nhưng gói *onboard* có thể được cài đặt thủ công để cung cấp bàn phím ảo đơn giản có thể sử dụng được trong mọi môi trường máy tính. Sau khi cài đặt, nó sẽ có sẵn dưới dạng một ứng dụng thông thường trong trình khởi chạy ứng dụng.

Hành vi của con trỏ cũng có thể được sửa đổi nếu việc nhấp và kéo chuột gây đau cho người dùng hoặc không thể thực hiện được vì bất kỳ lý do nào khác. Ví dụ: nếu người dùng không thể nhấp

vào nút chuột đủ nhanh để kích hoạt thao tác nhấp đúp, khoảng thời gian nhấn nút chuột lần thứ hai để nhấp đúp có thể tăng lên ở *Tùy chọn Chuột* (Mouse Preferences) trong cửa sổ cấu hình hệ thống.

Nếu người dùng không thể nhấn một hoặc bất kỳ nút nào trong các nút của chuột thì việc nhấp chuột có thể được mô phỏng bằng các kỹ thuật khác nhau. Trong phần *Hỗ trợ nhấp* (Click Assist) của *Trợ năng phổ quát Gnome*, tùy chọn *Mô phỏng thao tác nhấp chuột phải* (Simulate a right mouse click) sẽ tạo ra một cú nhấp chuột phải nếu người dùng nhấn và giữ nút chuột trái. Khi bật tùy chọn *Mô phỏng thao tác nhấp chuột bằng cách giữ chuột* (Simulate clicking by hovering), thao tác nhấp chuột sẽ được kích hoạt khi người dùng giữ yên chuột. Trong KDE, ứng dụng *KMouseTool* sẽ cung cấp những tính năng tương tự để hỗ trợ thao tác chuột.

Hạn chế về Thị giác

Người dùng bị suy giảm thị lực vẫn có thể sử dụng màn hình điều khiển để tương tác với máy tính. Tùy thuộc vào nhu cầu của người dùng, nhiều điều chỉnh trực quan có thể được thực hiện để cải thiện các chi tiết khó nhìn rõ trên màn hình đồ họa tiêu chuẩn.

Phần *Xem* (Seeing) của Gnome trong cài đặt *Trợ năng phổ quát* có cung cấp các tùy chọn có thể trợ giúp những người bị suy giảm thị lực:

Độ tương phản cao (High Contrast)

sẽ làm cho các cửa sổ và nút bấm trở nên dễ nhìn hơn bằng cách vẽ chúng bằng màu sắc sắc nét hơn.

Cỡ chữ lớn (Large Text)

sẽ phóng to kích thước phông chữ màn hình tiêu chuẩn.

Kích cỡ con trỏ (Cursor Size)

cho phép chọn con trỏ chuột lớn hơn, giúp định vị trên màn hình dễ dàng hơn.

Một số điều chỉnh này không liên quan chặt chẽ đến các tính năng trợ năng, do đó chúng có thể được tìm thấy trong phần giao diện của tiện ích cấu hình được cung cấp bởi các môi trường máy tính khác. Người dùng gặp khó khăn trong việc phân biệt các yếu tố trực quan có thể chọn các chủ đề có độ tương phản cao để giúp xác định các nút, cửa sổ chồng chéo, v.v. một cách dễ dàng hơn.

Nếu việc điều chỉnh giao diện không đủ để cải thiện khả năng đọc trên màn hình thì người dùng có thể sử dụng chương trình phóng to màn hình để phóng to các phần của màn hình. Tính năng này được gọi là *Zoom* trong cài đặt *Trợ năng phổ quát* của Gnome; trong đó, các tùy chọn như tỷ lệ phóng đại, vị trí của kính lúp và điều chỉnh màu sắc có thể được tùy chỉnh.

Trong KDE, chương trình *KMagnifier* cũng cung cấp các tính năng tương tự nhưng nó sẽ có sẵn dưới dạng một ứng dụng thông thường thông qua trình khởi chạy ứng dụng. Các môi trường máy tính khác có thể sẽ cung cấp các kính lúp màn hình riêng. Ví dụ: Xfce sẽ phóng to và thu nhỏ màn hình bằng cách xoay con lăn chuột trong khi phím **Alt** đang ở trạng giữ.

Cuối cùng, những người dùng không có tùy chọn về giao diện đồ họa có thể sử dụng một *trình đọc màn hình* để tương tác với máy tính. Bất kể môi trường máy tính được chọn là gì, trình đọc màn hình phổ biến nhất dành cho hệ thống Linux là *Orca* và nó thường được cài đặt theo mặc định trong hầu hết các bản phân phối. Orca tạo ra một giọng nói tổng hợp để báo cáo các sự kiện trên màn hình và đọc văn bản dưới con trỏ chuột. Orca cũng làm việc với *màn hình chữ nổi có thể làm mới* - các thiết bị đặc biệt hiển thị các ký tự chữ nổi bằng các nốt nhỏ có thể cảm nhận được bằng đầu ngón tay. Không phải tất cả các ứng dụng dành cho máy tính đều được điều chỉnh để tương thích hoàn toàn với mọi trình đọc màn hình và không phải người dùng nào cũng có thể sử dụng chúng một cách dễ dàng. Vì vậy, quan trọng nhất là càng nhiều giải pháp đọc màn hình được cung cấp càng tốt để người dùng có thể lựa chọn.

Bài tập Hướng dẫn

1. Tính năng trợ năng nào có thể giúp người dùng chuyển đổi giữa các cửa sổ đang mở bằng bàn phím khi người dùng không thể nhấn các phím **Alt** và **Tab** cùng một lúc?

2. Những người dùng bị run tay thường vô tình làm ảnh hưởng đến việc gõ phím của họ. Tính năng trợ năng *Phím nảy* có thể hỗ trợ những người dùng này như thế nào?

3. Cử chỉ kích hoạt phổ biến nhất cho tính năng trợ năng *Phím dính* là gì?

Bài tập Mở rộng

1. Các tính năng trợ năng có thể không được cung cấp bởi một ứng dụng duy nhất và có thể sẽ khác nhau tùy theo môi trường máy tính. Trong KDE, ứng dụng nào sẽ giúp những người bị chấn thương tay bằng cách nhấp chuột bất cứ khi nào con trỏ chuột tạm dừng trong một thời gian ngắn?

2. Những khía cạnh nào của môi trường đồ họa có thể được sửa đổi để giúp mọi người đọc văn bản trên màn hình một cách dễ dàng hơn?

3. Ứng dụng *Orca* có thể giúp người dùng khiếm thị tương tác với môi trường máy tính bằng những cách nào?

Tóm tắt

Bài học này đã nói về các tính năng trợ năng tổng quát có sẵn trong hệ thống Linux. Tất cả các môi trường máy tính chính (đặc biệt là Gnome và KDE) đều cung cấp nhiều ứng dụng tích hợp và ứng dụng của bên thứ ba để hỗ trợ những người khiếm thị hoặc suy giảm khả năng chuyển động. Bài học đã đi qua các chủ đề sau:

- Cách thay đổi cài đặt trợ năng.
- Các cách khác để sử dụng bàn phím và chuột.
- Các cải tiến máy tính dành cho người khiếm thị.

Các khái niệm và quy trình được đề cập tới là:

- Cài đặt trợ năng bàn phím: Phím dính, Phím chậm, Phím nảy.
- Tạo các thao tác chuột nhân tạo.
- Bàn phím trên màn hình.
- Cài đặt trực quan để nâng cao khả năng đọc.
- Các chủ đề của máy tính để bàn có độ tương phản cao/cỡ chữ lớn.
- Kính lúp màn hình.
- Trình đọc màn hình Orca.

Đáp án Bài tập Hướng dẫn

1. Tính năng trợ năng nào có thể giúp người dùng chuyển đổi giữa các cửa sổ đang mở bằng bàn phím khi người dùng không thể nhấn các phím `Alt` và `Tab` cùng một lúc?

Tính năng Phím dính cho phép người dùng gõ phím tắt, mỗi lần một phím.

+. Những người dùng bị run tay thường vô tình làm ảnh hưởng đến việc gõ phím của họ. Tính năng trợ năng *Phím nảy* có thể hỗ trợ những người dùng này như thế nào? Khi bật Phím nảy, một lần nhấn phím mới sẽ chỉ được chấp nhận sau một khoảng thời gian nhất định trôi qua kể từ lần nhấn phím cuối cùng.

1. Cử chỉ kích hoạt phổ biến nhất cho tính năng trợ năng *Phím dính* là gì?

Nếu Cử chỉ kích hoạt được bật, tính năng Phím dính sẽ được kích hoạt sau khi nhấn phím `Shift` năm lần liên tiếp.

Đáp án Bài tập Mở rộng

1. Các tính năng trợ năng có thể không được cung cấp bởi một ứng dụng duy nhất và có thể sẽ khác nhau tùy theo môi trường máy tính. Trong KDE, ứng dụng nào sẽ giúp những người bị chấn thương tay bằng cách nhấp chuột bất cứ khi nào con trỏ chuột tạm dừng trong một thời gian ngắn?

Ứng dụng *KMouseTool*.

2. Những khía cạnh nào của môi trường đồ họa có thể được sửa đổi để giúp mọi người đọc văn bản trên màn hình một cách dễ dàng hơn?

Đặt kích thước phông chữ màn hình lớn trong cấu hình máy tính sẽ giúp đọc tất cả các văn bản trên màn hình một cách dễ dàng hơn.

3. Ứng dụng *Orca* có thể giúp người dùng khiếm thị tương tác với môi trường máy tính bằng những cách nào?

Orca là trình đọc màn hình tạo ra một giọng nói tổng hợp để báo cáo các sự kiện trên màn hình và đọc văn bản dưới con trỏ chuột. Nó cũng hoạt động với các thiết bị được gọi là *màn hình chữ nổi có thể làm mới*, vì vậy nên người dùng có thể nhận dạng văn bản bằng các mẫu xúc giác.



Chủ đề 107: Tác vụ Quản trị



107.1 Quản lý Tài khoản Nhóm và Người dùng cũng như các Tập hệ thống liên quan

Tham khảo các mục tiêu LPI

[LPIC-1 version 5.0, Exam 102, Objective 107.1](#)

Khối lượng

5

Các lĩnh vực kiến thức chính

- Thêm, sửa đổi và xóa người dùng và nhóm.
- Quản lý thông tin người dùng/nhóm trong cơ sở dữ liệu mật khẩu/nhóm.
- Tạo và quản lý các tài khoản giới hạn và có mục đích đặc biệt.

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- `/etc/passwd`
- `/etc/shadow`
- `/etc/group`
- `/etc/skel/`
- `chage`
- `getent`
- `groupadd`
- `groupdel`
- `groupmod`
- `passwd`

- `useradd`
- `userdel`
- `usermod`



107.1 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	107 Các Tác vụ Quản trị
Mục tiêu:	107.1 Quản lý Tài khoản Nhóm và Người dùng cũng như các Tập hệ thống liên quan
Bài:	1 trên 2

Giới thiệu

Quản trị người dùng và nhóm là một phần rất quan trọng trong công việc của mọi quản trị viên hệ thống. Các bản phân phối Linux hiện đại triển khai các giao diện đồ họa cho phép người dùng quản lý tất cả các hoạt động liên quan đến khía cạnh quan trọng này một cách nhanh chóng và dễ dàng. Các giao diện này sẽ khác nhau về bố cục đồ họa nhưng giống nhau về tính năng. Với những công cụ này, người dùng có thể xem, chỉnh sửa, thêm và xóa người dùng và các nhóm cục bộ. Tuy nhiên, để quản lý nâng cao hơn, người dùng cần phải làm việc thông qua dòng lệnh.

Thêm Tài khoản Người dùng

Trong Linux, chúng ta có thể thêm tài khoản người dùng mới bằng lệnh `useradd`. Ví dụ: khi hoạt động với quyền gốc, chúng ta có thể tạo một tài khoản người dùng mới có tên `michael` với một cài đặt mặc định bằng cách sử dụng lệnh sau:

```
# useradd michael
```

Khi chạy lệnh `useradd`, thông tin người dùng và nhóm được lưu trữ trong cơ sở dữ liệu nhóm và mật khẩu sẽ được cập nhật cho tài khoản người dùng mới được tạo này và nếu được chỉ định, thư mục chính của người dùng mới cũng sẽ được tạo. Một nhóm có cùng tên với tài khoản người dùng mới cũng sẽ được tạo.

Một khi đã tạo người dùng mới, chúng ta có thể đặt mật khẩu của người dùng đó bằng lệnh `passwd`. Ta có thể xem lại ID người dùng (UID), ID nhóm (GID) của nó và các nhóm mà nó thuộc về thông qua các lệnh `id` và `groups`.

```
# passwd michael
Changing password for user michael.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
# id michael
uid=1000(michael) gid=100(michael) groups=100(michael)
# groups michael
michael : michael
```

NOTE

Hãy nhớ rằng bất kỳ người dùng nào cũng có thể xem lại UID, GID của họ và các nhóm mà họ thuộc về chỉ đơn giản bằng cách sử dụng lệnh `id` và `groups` mà không cần đối số. Họ cũng có thể thay đổi mật khẩu của mình bằng lệnh `passwd`. Tuy nhiên, chỉ những người dùng có quyền gốc mới có thể thay đổi mật khẩu của *mọi* người dùng.

Các tùy chọn quan trọng nhất áp dụng cho lệnh `useradd` là:

-c

Tạo một tài khoản người dùng mới với các chú thích tùy chỉnh (ví dụ: tên đầy đủ của người dùng).

-d

Tạo một tài khoản người dùng mới với thư mục chính tùy chỉnh.

-e

Tạo một tài khoản người dùng mới bằng cách đặt ngày cụ thể mà tài khoản đó sẽ bị vô hiệu hóa.

-f

Tạo một tài khoản người dùng mới bằng cách đặt số ngày mà sau khi mật khẩu hết hạn người dùng nên cập nhật mật khẩu (nếu không tài khoản sẽ bị vô hiệu hóa).

-g

Tạo một tài khoản người dùng mới với một GID cụ thể.

-G

Tạo một tài khoản người dùng mới bằng cách thêm nó vào nhiều nhóm phụ.

-k

Tạo một tài khoản người dùng mới bằng cách sao chép các tệp skeleton từ một thư mục tùy chỉnh cụ thể (tùy chọn này chỉ hợp lệ nếu tùy chọn `-m` hoặc `--create-home` được chỉ định).

-m

Tạo một tài khoản người dùng mới với thư mục chính của nó (nếu nó không tồn tại).

-M

Tạo một tài khoản người dùng mới mà không có thư mục chính.

-s

Tạo một tài khoản người dùng mới với một vỏ đăng nhập cụ thể.

-u

Tạo một tài khoản người dùng mới với một UID cụ thể.

Hãy xem các trang hướng dẫn của lệnh `useradd` để biết danh sách tùy chọn đầy đủ.

Sửa đổi Tài khoản Người dùng

Đôi khi chúng ta sẽ cần thay đổi thuộc tính của một tài khoản người dùng hiện có, chẳng hạn như tên đăng nhập, vỏ đăng nhập, ngày hết hạn mật khẩu, v.v. Trong những trường hợp như vậy, chúng ta sẽ sử dụng lệnh `usermod`.

```
# usermod -s /bin/tcsh michael
# usermod -c "Michael User Account" michael
```

Cũng giống như lệnh `useradd`, lệnh `usermod` cũng yêu cầu quyền gốc.

Trong các ví dụ trên, vỏ đăng nhập của `michael` sẽ được thay đổi trước tiên và sau đó một mô tả ngắn gọn sẽ được thêm vào tài khoản người dùng này. Hãy nhớ rằng chúng ta có thể sửa đổi nhiều thuộc tính cùng một lúc và chỉ định chúng bằng một lệnh duy nhất.

Các tùy chọn quan trọng nhất áp dụng cho lệnh `usermod` là:

-c

Thêm một chú thích ngắn gọn vào tài khoản người dùng được chỉ định.

-d

Thay đổi thư mục chính của tài khoản người dùng được chỉ định. Khi được sử dụng với tùy chọn `-m`, nội dung của thư mục chính hiện tại sẽ được chuyển sang thư mục chính mới, thư mục này sẽ được tạo nếu nó chưa tồn tại.

-e

Đặt ngày hết hạn của tài khoản người dùng được chỉ định.

-f

Đặt số ngày sau khi mật khẩu hết hạn mà người dùng nên cập nhật mật khẩu (nếu không tài khoản sẽ bị vô hiệu hóa).

-g

Thay đổi nhóm chính của tài khoản người dùng được chỉ định (nhóm phải tồn tại).

-G

Thêm nhóm phụ vào tài khoản người dùng được chỉ định. Mỗi nhóm đều phải tồn tại và phải được phân tách với nhóm tiếp theo bằng dấu phẩy và không có khoảng trắng ở giữa. Nếu được sử dụng một mình, tùy chọn này sẽ xóa tất cả các nhóm hiện có mà người dùng thuộc về. Nếu được sử dụng với tùy chọn `-a`, nó sẽ chỉ đơn giản là thêm các nhóm phụ mới vào các nhóm hiện có.

-l

Thay đổi tên đăng nhập của tài khoản người dùng đã chỉ định.

-L

Khóa tài khoản người dùng được chỉ định. Thao tác này sẽ đặt dấu chấm than trước mật khẩu được mã hóa trong tệp `/etc/shadow`, từ đó vô hiệu hóa quyền truy cập bằng mật khẩu cho người dùng đó.

-s

Thay đổi vỏ đăng nhập của tài khoản người dùng đã chỉ định.

-u

Thay đổi UID của tài khoản người dùng được chỉ định.

-U

Mở khóa tài khoản người dùng được chỉ định. Thao tác này sẽ xóa dấu chấm than phía trước mật khẩu được mã hóa bằng tệp `/etc/shadow`.

Hãy xem các trang hướng của lệnh `usermod` để biết danh sách tùy chọn đầy đủ.

TIP

Hãy nhớ rằng khi thay đổi tên đăng nhập của một tài khoản người dùng, bạn nên đổi tên thư mục chính của người dùng đó và các mục khác liên quan đến người dùng như tệp bộ đệm thư. Cũng nên nhớ rằng khi thay đổi UID của tài khoản người dùng, bạn cũng nên sửa quyền sở hữu các tệp và thư mục bên ngoài thư mục chính của người dùng (ID người dùng sẽ được thay đổi tự động cho hộp thư của người dùng và cho tất cả các tệp do người dùng sở hữu và nằm trong thư mục chính của người dùng).

Xóa Tài khoản Người dùng

Nếu muốn xóa tài khoản người dùng, chúng ta có thể sử dụng lệnh `userdel`. Đặc biệt, lệnh này sẽ cập nhật thông tin được lưu trữ trong cơ sở dữ liệu tài khoản, xóa tất cả các mục liên quan đến người dùng được chỉ định. Tùy chọn `-r` cũng sẽ xóa thư mục chính của người dùng và toàn bộ nội dung của nó cùng với bộ đệm thư. Các tệp khác nằm ở những nơi khác phải được tìm kiếm và xóa theo cách thủ công.

```
# userdel -r michael
```

Đối với `useradd` và `usermod`, người dùng cần có quyền gốc để xóa tài khoản người dùng.

Thêm, sửa đổi và xóa Nhóm

Cũng giống như việc quản lý người dùng, chúng ta có thể thêm, sửa đổi và xóa nhóm bằng cách sử dụng lệnh `groupadd`, `groupmod` và `groupdel` với quyền gốc. Nếu muốn tạo một nhóm mới có tên `developer`, chúng ta có thể chạy lệnh sau:

```
# groupadd -g 1090 developer
```

Tùy chọn `-g` của lệnh này sẽ tạo một nhóm với một GID cụ thể.

WARNING

Hãy nhớ rằng khi thêm một tài khoản người dùng mới, nhóm chính và nhóm phụ mà tài khoản đó thuộc về *phải* tồn tại trước khi khởi chạy lệnh `useradd`.

Sau này, nếu muốn đổi tên nhóm từ `developer` thành `web-developer` và thay đổi GID của nhóm, chúng ta có thể chạy lệnh như sau:

```
# groupmod -n web-developer -g 1050 developer
```

TIP

Hãy nhớ rằng nếu thay đổi GID bằng tùy chọn `-g`, bạn nên thay đổi GID của tất cả các tệp và thư mục phải tiếp tục thuộc về nhóm này.

Cuối cùng, nếu muốn xóa nhóm `web-developer`, ta có thể chạy lệnh như sau:

```
# groupdel web-developer
```

Chúng ta không thể xóa nhóm nếu đó là nhóm chính của tài khoản người dùng. Do đó, ta sẽ phải xóa người dùng trước khi xóa nhóm. Đối với người dùng, nếu ta xóa một nhóm, các tệp thuộc nhóm đó vẫn sẽ còn trong hệ thống tệp và sẽ không bị xóa hoặc gán cho một nhóm khác.

Thư mục Skeleton

Khi thêm một tài khoản người dùng mới hay thậm chí là tạo thư mục chính cho nó, thư mục chính mới được tạo sẽ chứa các tệp và thư mục được sao chép từ thư mục skeleton (theo mặc định `/etc/skel`). Ý tưởng đằng sau việc này rất đơn giản: một quản trị viên hệ thống muốn thêm người dùng mới có cùng tệp và thư mục vào thư mục chính của họ. Do đó, nếu muốn tùy chỉnh các tệp và thư mục được tạo tự động trong thư mục chính của tài khoản người dùng mới, chúng ta sẽ phải thêm các tệp và thư mục mới này vào thư mục skeleton.

TIP

Hãy lưu ý rằng nếu muốn liệt kê tất cả các tệp và thư mục trong thư mục skeleton, bạn sẽ phải sử dụng lệnh `ls -al`.

Tệp `/etc/login.defs`

Trong Linux, tệp `/etc/login.defs` sẽ chỉ định các tham số cấu hình kiểm soát việc tạo người dùng và nhóm. Ngoài ra, các lệnh hiển thị trong phần trước sẽ lấy giá trị mặc định từ tệp này.

Các chỉ thị quan trọng nhất là:

UID_MIN và **UID_MAX**

Phạm vi ID người dùng có thể được chỉ định cho người dùng thông thường mới.

GID_MIN và **GID_MAX**

Phạm vi ID nhóm có thể được gán cho các nhóm thông thường mới.

CREATE_HOME

Chỉ định xem có nên tạo thư mục chính theo mặc định cho người dùng mới hay không.

USERGROUPS_ENAB

Chỉ định xem hệ thống có nên tạo nhóm mới theo mặc định cho mỗi tài khoản người dùng mới có cùng tên với người dùng hay không, và liệu việc xóa tài khoản người dùng có nên xóa cả nhóm chính của người dùng hay không nếu nhóm đó không còn chứa thành viên.

MAIL_DIR

Thư mục bộ đệm thư.

PASS_MAX_DAYS

Số ngày tối đa mà mật khẩu có thể được sử dụng.

PASS_MIN_DAYS

Số ngày tối thiểu cho phép giữa các lần thay đổi mật khẩu.

PASS_MIN_LEN

Độ dài mật khẩu tối thiểu được chấp nhận.

PASS_WARN_AGE

Số ngày cảnh báo trước khi mật khẩu hết hạn.

TIP

Khi quản lý người dùng và nhóm, hãy luôn kiểm tra tệp này để xem và thay đổi hành vi mặc định của hệ thống nếu cần.

Lệnh `passwd`

Lệnh này chủ yếu được sử dụng để thay đổi mật khẩu của người dùng. Như đã được mô tả trước đó, bất kỳ người dùng nào cũng có thể thay đổi mật khẩu của riêng mình, nhưng chỉ có siêu người dùng mới có thể thay đổi mật khẩu của *mọi* người dùng. Điều này là do lệnh `passwd` có bộ bit SUID (chữ `s` ở vị trí cờ thực thi dành cho chủ sở hữu), có nghĩa là nó sẽ thực thi với các đặc quyền của chủ sở hữu tệp (tức quyền gốc).

```
# ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 42096 mag 17 2015 /usr/bin/passwd
```

Tùy thuộc vào các tùy chọn `passwd` được sử dụng, chúng ta có thể kiểm soát các khía cạnh cụ thể của quá trình lão hóa mật khẩu:

-d

Xóa mật khẩu của tài khoản người dùng (từ đó vô hiệu hóa người dùng).

-e

Buộc tài khoản người dùng thay đổi mật khẩu.

-i

Đặt số ngày không hoạt động sau khi mật khẩu hết hạn mà trong thời gian đó người dùng nên cập nhật mật khẩu (nếu không tài khoản sẽ bị vô hiệu hóa).

-l

Khóa tài khoản người dùng (mật khẩu được mã hóa với tiền tố là dấu chấm than trong tệp `/etc/shadow`).

-n

Đặt thời gian tồn tại tối thiểu của mật khẩu.

-S

Xuất thông tin về trạng thái mật khẩu của một tài khoản người dùng cụ thể.

-u

Mở khóa tài khoản người dùng (dấu chấm than sẽ được xóa khỏi trường mật khẩu trong tệp `/etc/shadow`).

-x

Đặt thời gian tồn tại tối đa của mật khẩu.

-w

Đặt số ngày cảnh báo trước khi mật khẩu hết hạn mà trong thời gian đó người dùng được cảnh báo rằng phải thay đổi mật khẩu.

NOTE

Các nhóm cũng có thể có mật khẩu; mật khẩu này có thể được đặt bằng lệnh `gpasswd`. Người dùng không phải là thành viên của nhóm nhưng biết mật khẩu của nhóm đó có thể tham gia tạm thời bằng lệnh `newgrp`. Hãy nhớ rằng `gpasswd` cũng được sử dụng để thêm, xóa người dùng khỏi một nhóm và thiết lập danh sách quản trị viên và thành viên thông thường của nhóm.

Lệnh `chage`

Lệnh này là viết tắt của “change age” và được sử dụng để thay đổi thông tin lão hoá mật khẩu của

người dùng. Lệnh `chage` bị giới hạn ở quyền gốc (ngoại trừ tùy chọn `-l` mà người dùng thông thường có thể sử dụng để liệt kê thông tin lão hoá mật khẩu của tài khoản của họ).

Các tùy chọn khác áp dụng cho lệnh `chage` là:

-d

Đặt lần thay đổi mật khẩu cuối cùng cho tài khoản người dùng.

-E

Đặt ngày hết hạn cho tài khoản người dùng.

-I

Đặt số ngày không hoạt động sau khi mật khẩu hết hạn mà trong thời gian đó người dùng nên cập nhật mật khẩu (nếu không tài khoản sẽ bị vô hiệu hóa).

-m

Đặt thời gian tồn tại tối thiểu của mật khẩu cho tài khoản người dùng.

-M

Đặt thời gian tồn tại tối đa của mật khẩu cho tài khoản người dùng.

-W

Đặt số ngày cảnh báo trước khi mật khẩu hết hạn mà trong thời gian đó người dùng được cảnh báo rằng phải thay đổi mật khẩu.

Bài tập Hướng dẫn

1. Đối với mỗi lệnh sau, hãy xác định mục đích tương ứng của nó:

<code>usermod -L</code>	
<code>passwd -u</code>	
<code>chage -E</code>	
<code>groupdel</code>	
<code>useradd -s</code>	
<code>groupadd -g</code>	
<code>userdel -r</code>	
<code>usermod -l</code>	
<code>groupmod -n</code>	
<code>useradd -m</code>	

2. Đối với mỗi lệnh `passwd` sau, hãy xác định lệnh `chage` tương ứng:

<code>passwd -n</code>	
<code>passwd -x</code>	
<code>passwd -w</code>	
<code>passwd -i</code>	
<code>passwd -S</code>	

3. Hãy giải thích chi tiết mục đích của các lệnh trong câu hỏi trước:

4. Bạn có thể sử dụng lệnh nào để khóa tài khoản người dùng? Và lệnh nào để mở khóa cho nó?

Bài tập Mở rộng

1. Giả sử bạn đang làm việc với quyền gốc, bằng cách sử dụng lệnh `groupadd`, hãy tạo nhóm `administrators` và `developers`.

2. Sau khi tạo các nhóm, hãy chạy lệnh sau: `useradd -G administrators,developers kevin`. Lệnh này đã thực hiện những hoạt động gì? (Giả sử rằng `CREATE_HOME` và `USERGROUPS_ENAB` trong `/etc/login.defs` được đặt thành `yes`).

3. Hãy tạo một nhóm mới có tên là `designers`, đổi tên thành `web-designers` và thêm nhóm mới này vào các nhóm phụ của tài khoản người dùng `kevin`. Hãy xác định tất cả các nhóm mà `kevin` thuộc về và ID của chúng.

4. Hãy chỉ xóa nhóm `developers` khỏi các nhóm phụ của `kevin`.

5. Hãy đặt mật khẩu cho tài khoản người dùng `kevin`.

6. Bằng cách sử dụng lệnh `chage`, trước tiên hãy kiểm tra ngày hết hạn của tài khoản người dùng `kevin`, sau đó đổi nó thành ngày 31 tháng 12 năm 2022. Bạn có thể sử dụng lệnh nào khác để thay đổi ngày hết hạn của tài khoản người dùng?

7. Hãy thêm tài khoản người dùng mới có tên `emma` với UID 1050 và đặt `administrators` làm nhóm chính và `developers` và `web-designers` làm nhóm phụ.

8. Hãy thay đổi vỏ đăng nhập của `emma` thành `/bin/sh`.

9. Hãy xóa tài khoản người dùng `emma` và `kevin` và các nhóm `administrators`, `developers` và `web-designers`.

Tóm tắt

Trong bài học này, chúng ta đã học về:

- Các nguyên tắc cơ bản về quản lý người dùng và nhóm trong Linux.
- Cách thêm, sửa đổi và xóa tài khoản người dùng.
- Cách thêm, sửa và xóa tài khoản nhóm.
- Duy trì thư mục skeleton.
- Chỉnh sửa tệp kiểm soát việc tạo người dùng và nhóm.
- Thay đổi mật khẩu của tài khoản người dùng.
- Thay đổi thông tin lão hóa mật khẩu của tài khoản người dùng.

Các tệp và lệnh sau đã được thảo luận trong bài học này:

useradd

Tạo một tài khoản người dùng mới.

usermod

Sửa đổi một tài khoản người dùng.

userdel

Xóa một tài khoản người dùng.

groupadd

Tạo một tài khoản nhóm mới.

groupmod

Sửa đổi một tài khoản nhóm.

groupdel

Xóa một tài khoản nhóm

passwd

Thay đổi mật khẩu của tài khoản người dùng và kiểm soát tất cả các khía cạnh của quá trình lão hóa mật khẩu.

chage

Thay đổi thông tin hết hạn của mật khẩu người dùng.

/etc/skel

Vị trí mặc định của thư mục skeleton.

/etc/login.defs

Tập kiểm soát việc tạo người dùng và nhóm và cung cấp các giá trị mặc định cho một số tham số tài khoản người dùng.

Đáp án Bài tập Hướng dẫn

1. Đối với mỗi lệnh sau, hãy xác định mục đích tương ứng của nó:

<code>usermod -L</code>	Khóa tài khoản người dùng
<code>passwd -u</code>	Mở khóa tài khoản người dùng
<code>chage -E</code>	Đặt ngày hết hạn cho tài khoản người dùng
<code>groupdel</code>	Xóa nhóm
<code>useradd -s</code>	Tạo một tài khoản người dùng mới với một vỏ đăng nhập cụ thể
<code>groupadd -g</code>	Tạo một nhóm mới với một GID cụ thể
<code>userdel -r</code>	Xóa tài khoản người dùng và tất cả các tệp trong thư mục chính của nó, bản thân thư mục chính và bộ đệm thư của người dùng
<code>usermod -l</code>	Thay đổi tên đăng nhập của tài khoản người dùng
<code>groupmod -n</code>	Thay đổi tên của nhóm
<code>useradd -m</code>	Tạo một tài khoản người dùng mới và thư mục chính của nó

2. Đối với mỗi lệnh `passwd` sau, hãy xác định lệnh `chage` tương ứng:

<code>passwd -n</code>	<code>chage -m</code>
<code>passwd -x</code>	<code>chage -M</code>
<code>passwd -w</code>	<code>chage -W</code>
<code>passwd -i</code>	<code>chage -I</code>
<code>passwd -S</code>	<code>chage -l</code>

3. Hãy giải thích chi tiết mục đích của các lệnh trong câu hỏi trước:

Trong Linux, bạn có thể sử dụng lệnh `passwd -n` (hoặc `chage -m`) để đặt số ngày tối thiểu giữa các lần thay đổi mật khẩu, lệnh `passwd -x` (hoặc `chage -M`) để đặt số ngày tối đa mà mật khẩu hợp lệ, lệnh `passwd -w` (hoặc `chage -W`) để đặt số ngày cảnh báo trước khi mật khẩu hết hạn, lệnh `passwd -i` (hoặc `chage -I`) để đặt số ngày không hoạt động mà trong thời gian đó người dùng nên thay đổi mật khẩu và lệnh `passwd -S` (hoặc `chage -l`) để hiển thị thông tin ngắn

gọn về mật khẩu của tài khoản người dùng.

4. Bạn có thể sử dụng lệnh nào để khóa tài khoản người dùng? Và lệnh nào để mở khóa cho nó?

Nếu muốn khóa một tài khoản người dùng, bạn có thể sử dụng một trong các lệnh sau: `usermod -L`, `usermod --lock` và `passwd -l`. Nếu muốn mở khóa, bạn có thể sử dụng `usermod -U`, `usermod --unlock` và `passwd -u`.

Đáp án Bài tập Mở rộng

- Giả sử bạn đang làm việc với quyền gốc, bằng cách sử dụng lệnh `groupadd`, hãy tạo nhóm `administrators` và `developers`

```
# groupadd administrators
# groupadd developers
```

- Sau khi tạo các nhóm, hãy chạy lệnh sau: `useradd -G administrators,developers kevin`. Lệnh này đã thực hiện những hoạt động gì? (Giả sử rằng `CREATE_HOME` và `USERGROUPS_ENAB` trong `/etc/login.defs` được đặt thành `yes`).

Lệnh đã thêm một người dùng mới là `kevin` vào danh sách người dùng trong hệ thống, tạo thư mục chính của nó (`CREATE_HOME` được đặt thành `"yes"` và do đó bạn có thể bỏ qua tùy chọn `-m`) và tạo một nhóm mới được đặt tên `kevin` (là nhóm chính của tài khoản người dùng này vì `USERGROUPS_ENAB` được đặt thành `"yes"`). Cuối cùng, các tệp và thư mục chứa trong thư mục `skeleton` đã được sao chép vào thư mục gốc của `kevin`.

- Hãy tạo một nhóm mới có tên là `designers`, đổi tên thành `web-designers` và thêm nhóm mới này vào các nhóm phụ của tài khoản người dùng `kevin`. Hãy xác định tất cả các nhóm mà `kevin` thuộc về và ID của chúng.

```
# groupadd designers
# groupmod -n web-designers designers
# usermod -a -G web-designers kevin
# id kevin
uid=1010(kevin) gid=1030(kevin)
groups=1030(kevin),1028(administrators),1029(developers),1031(web-designers)
```

- Hãy chỉ xóa nhóm `developers` khỏi các nhóm phụ của `kevin`.

```
# usermod -G administrators,web-designers kevin
# id kevin
uid=1010(kevin) gid=1030(kevin) groups=1030(kevin),1028(administrators),1031(web-designers)
```

Lệnh `usermod` không có tùy chọn chỉ xóa một nhóm; do đó, bạn cần chỉ định tất cả các nhóm phụ mà người dùng thuộc về.

- Hãy đặt mật khẩu cho tài khoản người dùng `kevin`.

```
# passwd kevin
Changing password for user kevin.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

6. Bằng cách sử dụng lệnh `chage`, trước tiên hãy kiểm tra ngày hết hạn của tài khoản người dùng `kevin`, sau đó đổi nó thành ngày 31 tháng 12 năm 2022. Bạn có thể sử dụng lệnh nào khác để thay đổi ngày hết hạn của tài khoản người dùng?

```
# chage -l kevin | grep "Account expires"
Account expires      : never
# chage -E 2022-12-31 kevin
# chage -l kevin | grep "Account expires"
Account expires      : dec 31, 2022
```

Lệnh `usermod` với tùy chọn `-e` sẽ tương đương với `chage -E`.

7. Hãy thêm tài khoản người dùng mới có tên `emma` với UID 1050 và đặt `administrators` làm nhóm chính và `developers` và `web-designers` làm nhóm phụ.

```
# useradd -u 1050 -g administrators -G developers,web-designers emma
# id emma
uid=1050(emma) gid=1028(administrators)
groups=1028(administrators),1029(developers),1031(web-designers)
```

8. Hãy thay đổi vỏ đăng nhập của `emma` thành `/bin/sh`.

```
# usermod -s /bin/sh emma
```

9. Hãy xóa tài khoản người dùng `emma` và `kevin` và các nhóm `administrators`, `developers` và `web-designers`.

```
# userdel -r emma
# userdel -r kevin
# groupdel administrators
# groupdel developers
# groupdel web-designers
```



107.1 Bài 2

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	107 Các Tác vụ Quản trị
Mục tiêu:	107.1 Quản lý Tài khoản Nhóm và Người dùng cũng như các Tập hệ thống liên quan
Bài:	2 trên 2

Giới thiệu

Các công cụ dòng lệnh được thảo luận trong bài học trước và các ứng dụng đồ họa được cung cấp bởi mỗi bản phân phối để thực hiện các tác vụ giống nhau sẽ cập nhật một loạt các tệp lưu trữ thông tin về người dùng và nhóm.

Các tệp này nằm trong thư mục `/etc/` và chúng là:

`/etc/passwd`

Một tệp gồm bảy trường được phân cách bằng dấu hai chấm chứa thông tin cơ bản về người dùng.

`/etc/group`

Một tệp gồm bốn trường được phân cách bằng dấu hai chấm chứa thông tin cơ bản về các nhóm.

`/etc/shadow`

Một tệp gồm chín trường được phân cách bằng dấu hai chấm chứa mật khẩu người dùng được

mã hóa.

/etc/gshadow

Một tập gồm bốn trường được phân cách bằng dấu hai chấm chứa mật khẩu nhóm được mã hóa.

Mặc dù bốn tập này đều ở dạng văn bản thuần túy nhưng chúng không nên được chỉnh sửa trực tiếp mà phải thông qua các công cụ được cung cấp bởi bản phân phối mà người dùng đang sử dụng.

/etc/passwd

Đây là một tập có thể được đọc trên phạm vi toàn cục có chứa danh sách người dùng, mỗi người dùng sẽ nằm trên một dòng riêng biệt. Mỗi dòng sẽ bao gồm bảy trường được phân cách bằng dấu hai chấm:

Tên người dùng

Tên được sử dụng khi người dùng đăng nhập vào hệ thống.

Mật khẩu

Mật khẩu được mã hóa (hoặc x nếu sử dụng mật khẩu ẩn).

ID người dùng (UID)

Số ID được gán cho người dùng trong hệ thống.

ID nhóm (GID)

Số ID nhóm chính của người dùng trong hệ thống.

GECOS

Trường chú thích tùy chọn được sử dụng để thêm thông tin bổ sung về người dùng (chẳng hạn như tên đầy đủ). Trường này có thể chứa nhiều mục được phân tách bằng dấu phẩy.

Thư mục chính

Đường dẫn tuyệt đối của thư mục chính của người dùng.

Vỏ

Đường dẫn tuyệt đối của chương trình được tự động khởi chạy khi người dùng đăng nhập vào hệ thống (thường là vỏ tương tác, chẳng hạn như `/bin/bash`).

/etc/group

Đây là một tệp có thể được đọc trên phạm vi toàn cục có chứa danh sách các nhóm, mỗi nhóm sẽ nằm trên một dòng riêng biệt. Mỗi dòng sẽ bao gồm bốn trường được phân cách bằng dấu hai chấm:

Tên nhóm

Tên của nhóm.

Mật khẩu nhóm

Mật khẩu được mã hóa của nhóm (hoặc x nếu sử dụng mật khẩu ẩn).

ID nhóm (GID)

Số ID được gán cho nhóm trong hệ thống.

Danh sách thành viên

Danh sách người dùng thuộc nhóm được phân cách bằng dấu phẩy (ngoại trừ những người có nhóm này là nhóm chính).

/etc/shadow

Đây là tệp chỉ có thể được đọc bởi siêu người dùng và những người dùng có quyền gốc bao gồm cả mật khẩu người dùng được mã hóa, mỗi mật khẩu nằm trên một dòng riêng biệt. Mỗi dòng sẽ bao gồm chín trường được phân cách bằng dấu hai chấm:

Tên người dùng

Tên được sử dụng khi người dùng đăng nhập vào hệ thống.

Mật khẩu được mã hóa

Mật khẩu được mã hóa của người dùng (nếu giá trị bắt đầu bằng !, tài khoản sẽ bị khóa).

Ngày thay đổi mật khẩu lần cuối

Ngày thay đổi mật khẩu cuối cùng: là số ngày kể từ ngày 01/01/1970 (giá trị 0 có nghĩa là người dùng phải thay đổi mật khẩu khi đăng nhập lần tiếp theo).

Tuổi mật khẩu tối thiểu

Số ngày tối thiểu phải trôi qua sau một lần thay đổi mật khẩu trước khi người dùng được phép tiếp tục thay đổi mật khẩu.

Tuổi mật khẩu tối đa

Số ngày tối đa phải trôi qua trước khi mật khẩu cần được thay đổi.

Thời gian cảnh báo mật khẩu

Số ngày trước khi mật khẩu hết hạn mà trong thời gian đó người dùng được cảnh báo phải thay đổi mật khẩu.

Thời gian không hoạt động của mật khẩu

Số ngày sau khi mật khẩu hết hạn mà người dùng phải cập nhật mật khẩu. Sau khoảng thời gian này, nếu người dùng không thay đổi mật khẩu, tài khoản sẽ bị vô hiệu hóa.

Ngày hết hạn tài khoản

Ngày tài khoản người dùng sẽ bị vô hiệu hóa (được biểu thị bằng số ngày kể từ ngày 01/01/1970; trường trống có nghĩa là tài khoản người dùng sẽ không bao giờ hết hạn).

Trường đặt riêng

Một trường được dành riêng để sử dụng trong tương lai.

/etc/gshadow

Đây là tệp chỉ có thể được đọc bởi siêu người dùng và những người dùng có quyền gốc bao gồm cả mật khẩu người dùng được mã hóa, mỗi mật khẩu nằm trên một dòng riêng biệt. Mỗi dòng sẽ bao gồm chín trường được phân cách bằng dấu hai chấm:

Tên nhóm

Tên của nhóm.

Mật khẩu được mã hóa

Mật khẩu được mã hóa cho nhóm (được sử dụng khi người dùng không phải là thành viên của nhóm và muốn tham gia nhóm bằng lệnh `newgrp` — nếu mật khẩu bắt đầu bằng `!` thì không ai được phép truy cập nhóm bằng `newgrp`).

Quản trị viên nhóm

Danh sách quản trị viên của nhóm được phân cách bằng dấu phẩy (họ có thể thay đổi mật khẩu của nhóm và có thể thêm hoặc xóa thành viên nhóm bằng lệnh `gpasswd`).

Thành viên nhóm

Danh sách các thành viên của nhóm được phân cách bằng dấu phẩy.

Lọc Cơ sở dữ liệu Mật khẩu và Nhóm

Chúng ta có thể sẽ thường xuyên phải xem lại thông tin về người dùng và nhóm được lưu trữ trong bốn tệp này và tìm kiếm các thông tin cụ thể. Để thực hiện tác vụ này, ta có thể sử dụng lệnh `grep` hoặc lệnh ghép giữa `cat` và `grep`.

```
# grep emma /etc/passwd
emma:x:1020:1020:User Emma:/home/emma:/bin/bash
# cat /etc/group | grep db-admin
db-admin:x:1050:grace, frank
```

Một cách khác để truy cập các cơ sở dữ liệu này là sử dụng lệnh `getent`. Nhìn chung, lệnh này sẽ hiển thị các mục từ cơ sở dữ liệu được thư viện *Chuyển đổi Dịch vụ Tên* (NSS) hỗ trợ và yêu cầu tên cơ sở dữ liệu cũng như một từ khóa tra cứu. Nếu không có đối số từ khóa nào được cung cấp, tất cả các mục trong cơ sở dữ liệu đã chỉ định sẽ được hiển thị (trừ khi cơ sở dữ liệu không hỗ trợ liệt kê). Mặt khác, nếu một hoặc nhiều đối số từ khóa được cung cấp, cơ sở dữ liệu sẽ được lọc tương ứng với những từ khóa đó.

```
# getent passwd emma
emma:x:1020:1020:User Emma:/home/emma:/bin/bash
# getent group db-admin
db-admin:x:1050:grace, frank
```

Lệnh `getent` không yêu cầu quyền gốc. Người dùng chỉ cần có khả năng đọc cơ sở dữ liệu mà họ muốn truy xuất thông tin.

NOTE

Hãy nhớ rằng `getent` chỉ có thể truy cập cơ sở dữ liệu được định cấu hình trong tệp `/etc/nsswitch.conf`.

Bài tập Hướng dẫn

1. Hãy quan sát kết quả đầu ra sau và trả lời các câu hỏi:

```
# cat /etc/passwd | grep '\(root\|mail\|catherine\|kevin\)'
root:x:0:0:root:/root:/bin/bash
mail:x:8:8:mail:/var/spool/mail:/sbin/nologin
catherine:x:1030:1025:User Chaterine:/home/catherine:/bin/bash
kevin:x:1040:1015:User Kevin:/home/kevin:/bin/bash
# cat /etc/group | grep '\(root\|mail\|db-admin\|app-developer\)'
root:x:0:
mail:x:8:
db-admin:x:1015:emma,grace
app-developer:x:1016:catherine,dave,christian
# cat /etc/shadow | grep '\(root\|mail\|catherine\|kevin\)'
root:$6$1u36Ipok$1jt8ooPMLewAhkQPf.1YgGopAB.jClT06ljsdczxvkLPkpi/amgp.zyfAN680zrLLp2avvpd
KA0llpssdfcPppOp:18015:0:99999:7:::
mail:*:18015:0:99999:7:::
catherine:$6$ABCD25jlld14hpPthEFGnnsEww1234yioMpliABCdef1f3478kAfhhAfgbAMjY1/BAeeAsl/FeE
dddKd12345g6kPACcik:18015:20:90:5:::
kevin:$6$DEFGabc123WrLp223fsvp0ddx3dbA7pPPc4LMaa123u6Lp02Lpvm123456pyphhh5ps012vbArL245.P
R1345kkA3Gas12P:18015:0:60:7:2:::
# cat /etc/gshadow | grep '\(root\|mail\|db-admin\|app-developer\)'
root:*:
mail:*:
db-admin:!:emma:emma,grace
app-developer:!:catherine,dave,christian
```

◦ ID người dùng (UID) và ID nhóm (GID) của `root` và `catherine` là gì?

◦ Tên nhóm chính của `kevin` là gì? Có thành viên nào khác trong nhóm này không?

◦ Vỏ nào được đặt cho `mail`? Nó có nghĩa là gì?

◦ Thành viên của nhóm `app-developer` là những ai? Thành viên nào trong số này là quản trị viên nhóm và thành viên nào là thành viên thông thường?

- Thời gian tồn tại tối thiểu của mật khẩu đối với catherine là bao lâu? Và thời gian tồn tại tối đa của mật khẩu là bao lâu?

- Khoảng thời gian không hoạt động của mật khẩu đối với kevin là bao lâu?

2. Theo quy ước, ID nào được gán cho tài khoản hệ thống và ID nào được gán cho người dùng thông thường?

3. Giả sử hệ thống của bạn sử dụng mật khẩu ẩn, làm cách nào để biết tài khoản người dùng mà trước đây có thể truy cập vào hệ thống bây giờ có bị khóa hay không?

Bài tập Mở rộng

1. Hãy tạo tài khoản người dùng có tên `christian` bằng cách sử dụng lệnh `useradd -m` và xác định ID người dùng (UID), ID nhóm (GID) và vỏ của tài khoản đó.

2. Hãy xác định tên nhóm chính của `christian`. Bạn có thể suy luận ra điều gì từ đây?

3. Bằng cách sử dụng lệnh `getent`, hãy xem thông tin lão hoá mật khẩu của tài khoản người dùng `christian`.

4. Hãy thêm nhóm `editor` vào các nhóm phụ của `christian`. Giả sử rằng nhóm này đã chứa `emma`, `dave` và `frank` là thành viên thông thường. Làm cách nào để có thể xác minh rằng nhóm này không có quản trị viên?

5. Hãy chạy lệnh `ls -l /etc/passwd /etc/group /etc/shadow /etc/gshadow` và mô tả đầu ra của nó về các quyền đối với tệp. Tệp nào trong số bốn tệp này bị ẩn vì lý do bảo mật (giả sử hệ thống của bạn sử dụng mật khẩu ẩn)?

Tóm tắt

Trong bài học này, chúng ta đã học về:

- Vị trí của các tệp lưu trữ thông tin về người dùng và nhóm.
- Quản lý thông tin người dùng và nhóm được lưu trữ trong cơ sở dữ liệu mật khẩu và nhóm.
- Lấy thông tin từ Cơ sở dữ liệu Mật khẩu và Nhóm.

Các tệp và lệnh sau đã được thảo luận trong bài học này:

/etc/passwd

Tệp chứa thông tin cơ bản về người dùng.

/etc/nhóm

Tệp chứa thông tin cơ bản về các nhóm.

/etc/shadow

Tệp chứa mật khẩu người dùng được mã hóa.

/etc/gshadow

Tệp chứa mật khẩu nhóm được mã hóa.

getent

Lọc Cơ sở dữ liệu Mật khẩu và Nhóm.

Đáp án Bài tập Hướng dẫn

1. Hãy quan sát kết quả đầu ra sau và trả lời các câu hỏi:

```
# cat /etc/passwd | grep '\(root\|mail\|catherine\|kevin\)'
root:x:0:0:root:/root:/bin/bash
mail:x:8:8:mail:/var/spool/mail:/sbin/nologin
catherine:x:1030:1025:User Chaterine:/home/catherine:/bin/bash
kevin:x:1040:1015:User Kevin:/home/kevin:/bin/bash
# cat /etc/group | grep '\(root\|mail\|db-admin\|app-developer\)'
root:x:0:
mail:x:8:
db-admin:x:1015:emma,grace
app-developer:x:1016:catherine,dave,christian
# cat /etc/shadow | grep '\(root\|mail\|catherine\|kevin\)'
root:$6$1u36Ipok$1jt8ooPMLewAhkQPf.1YgGopAB.jClT06ljsdczvxkLPkpi/amgp.zyfAN680zrLLp2avvpd
KA0llpssdfcPppOp:18015:0:99999:7:::
mail:*:18015:0:99999:7:::
catherine:$6$ABCD25jlld14hpPthEFGnnsEwW1234yioMpliABCdef1f3478kAfhhAfgbAMjY1/BAeeAsl/FeE
dddKd12345g6kPACcik:18015:20:90:5:::
kevin:$6$DEFGabc123WrLp223fsvp0ddx3dbA7pPPc4LMaa123u6Lp02Lpvm123456pyphhh5ps012vbArL245.P
R1345kkA3Gas12P:18015:0:60:7:2:::
# cat /etc/gshadow | grep '\(root\|mail\|db-admin\|app-developer\)'
root*::
mail*::
db-admin!:emma:emma,grace
app-developer!:catherine,dave,christian
```

- ID người dùng (UID) và ID nhóm (GID) của `root` và `catherine` là gì?

UID và GID của `root` là 0 và 0, trong khi UID và GID của `catherine` là 1030 và 1025.

- Tên nhóm chính của `kevin` là gì? Có thành viên nào khác trong nhóm này không?

Tên nhóm là `db-admin`. Ngoài ra `emma` và `grace` cũng nằm trong nhóm này.

- Vỏ nào được đặt cho `mail`? Nó có nghĩa là gì?

`mail` là tài khoản người dùng hệ thống và vỏ của nó là `/sbin/nologin`. Trên thực tế, các tài khoản người dùng hệ thống như `mail`, `ftp`, `news` và `daemon` được sử dụng để thực hiện các tác vụ quản trị và do đó việc đăng nhập thông thường nên bị chặn đối với các tài khoản này. Đây là lý do tại sao vỏ thường được đặt thành `/sbin/nologin` hoặc `/bin/false`.

- Thành viên của nhóm `app-developer` là những ai? Thành viên nào trong số này là quản trị viên nhóm và thành viên nào là thành viên thông thường?

Các thành viên là `catherine`, `dave` và `christian` và họ đều là những thành viên thông thường.

- Thời gian tồn tại tối thiểu của mật khẩu đối với `catherine` là bao lâu? Và thời gian tồn tại tối đa của mật khẩu là bao lâu?

Thời gian tồn tại của mật khẩu tối thiểu là 20 ngày, còn thời gian tồn tại tối đa là 90 ngày.

- Khoảng thời gian không hoạt động của mật khẩu đối với `kevin` là bao lâu?

Thời gian mật khẩu không hoạt động là 2 ngày. Trong thời gian này, `kevin` nên cập nhật mật khẩu; nếu không, tài khoản sẽ bị vô hiệu hóa.

2. Theo quy ước, ID nào được gán cho tài khoản hệ thống và ID nào được gán cho người dùng thông thường?

Tài khoản hệ thống thường có UID nhỏ hơn 100 hoặc từ 500 đến 1000, trong khi người dùng thông thường sẽ có UID bắt đầu từ 1000 (mặc dù một số hệ thống cũ có thể bắt đầu đánh số từ 500). Người dùng `root` có UID 0. Hãy nhớ rằng các giá trị `UID_MIN` và `UID_MAX` trong `/etc/login.defs` sẽ xác định phạm vi UID được sử dụng để tạo người dùng thông thường. Theo quan điểm của LPI Linux Essentials và LPIC-1, tài khoản hệ thống sẽ có UID nhỏ hơn 1000 và người dùng thông thường sẽ có UID lớn hơn 1000.

3. Giả sử hệ thống của bạn sử dụng mật khẩu ẩn, làm cách nào để biết tài khoản người dùng mà trước đây có thể truy cập vào hệ thống bây giờ có bị khóa hay không?

Khi mật khẩu ẩn được sử dụng, trường thứ hai trong `/etc/passwd` sẽ chứa ký tự `x` cho mỗi tài khoản người dùng vì mật khẩu người dùng được mã hóa sẽ được lưu trữ trong `/etc/shadow`. Đặc biệt, mật khẩu được mã hóa của tài khoản người dùng sẽ được lưu trữ trong trường thứ hai của tệp này và nếu nó bắt đầu bằng dấu chấm than thì tài khoản sẽ bị khóa.

Đáp án Bài tập Mở rộng

1. Hãy tạo tài khoản người dùng có tên `christian` bằng cách sử dụng lệnh `useradd -m` và xác định ID người dùng (UID), ID nhóm (GID) và vỏ của tài khoản đó.

```
# useradd -m christian
# cat /etc/passwd | grep christian
christian:x:1050:1060::/home/christian:/bin/bash
```

UID và GID của `christian` lần lượt là 1050 và 1060 (trường thứ ba và thứ tư trong `/etc/passwd`). `/bin/bash` là tệp vỏ cho tài khoản người dùng này (trường thứ bảy trong `/etc/passwd`).

2. Hãy xác định tên nhóm chính của `christian`. Bạn có thể suy luận ra điều gì từ đây?

```
# cat /etc/group | grep 1060
christian:x:1060:
```

Tên nhóm chính của `christian` là `christian` (trường đầu tiên trong `/etc/group`). Do đó, `USERGROUPS_ENAB` trong `/etc/login.defs` sẽ được đặt thành "Yes" để `useradd` tạo theo mặc định một nhóm có cùng tên với tài khoản người dùng.

3. Bằng cách sử dụng lệnh `getent`, hãy xem thông tin lão hoá mật khẩu của tài khoản người dùng `christian`.

```
# getent shadow christian
christian:!:18015:0:99999:7:::
```

Tài khoản người dùng `christian` không được đặt mật khẩu và hiện đã bị khóa (trường thứ hai trong `/etc/shadow` chứa dấu chấm than). Tài khoản người dùng này không có tuổi mật khẩu tối thiểu và tối đa (trường thứ tư và thứ năm trong `/etc/shadow` được đặt thành 0 và 99999 ngày), trong khi thời gian cảnh báo mật khẩu được đặt thành 7 ngày (trường thứ sáu trong `/etc/shadow`). Cuối cùng, không có khoảng thời gian không hoạt động (trường thứ bảy trong `/etc/shadow`) và tài khoản sẽ không bao giờ hết hạn (trường thứ tám trong `/etc/shadow`).

4. Hãy thêm nhóm `editor` vào các nhóm phụ của `christian`. Giả sử rằng nhóm này đã chứa `emma`, `dave` và `frank` là thành viên thông thường. Làm cách nào để có thể xác minh rằng nhóm này không có quản trị viên?

```
# cat /etc/group | grep editor
editor:x:1100:emma,dave,frank
# usermod -a -G editor christian
# cat /etc/group | grep editor
editor:x:1100:emma,dave,frank,christian
# cat /etc/gshadow | grep editor
editor:!:emma,dave,frank,christian
```

Trường thứ ba và thứ tư trong `/etc/gshadow` có chứa quản trị viên và thành viên thông thường của nhóm được chỉ định. Vì trường thứ ba của `editor` trống nên không có quản trị viên nào cho nhóm này (`emma`, `dave`, `frank` và `christian` đều là những thành viên thông thường).

5. Hãy chạy lệnh `ls -l /etc/passwd /etc/group /etc/shadow /etc/gshadow` và mô tả đầu ra của nó về các quyền đối với tệp. Tệp nào trong số bốn tệp này bị ẩn vì lý do bảo mật (giả sử hệ thống của bạn sử dụng mật khẩu ẩn)?

```
# ls -l /etc/passwd /etc/group /etc/shadow /etc/gshadow
-rw-r--r-- 1 root root 853 mag 1 08:00 /etc/group
-rw-r----- 1 root shadow 1203 mag 1 08:00 /etc/gshadow
-rw-r--r-- 1 root root 1354 mag 1 08:00 /etc/passwd
-rw-r----- 1 root shadow 1563 mag 1 08:00 /etc/shadow
```

Các tệp `/etc/passwd` và `/etc/group` có thể được đọc trên phạm vi toàn cục và bị ẩn vì lý do bảo mật. Khi mật khẩu ẩn được sử dụng, bạn có thể thấy `x` trong trường thứ hai của các tệp này vì mật khẩu được mã hóa cho người dùng và nhóm được lưu trữ trong `/etc/shadow` và `/etc/gshadow` và chỉ có thể đọc được bởi siêu người dùng và các thành viên thuộc nhóm `shadow`.



107.2 Tự động hóa các Tác vụ Quản trị Hệ thống bằng cách lập lịch trình công việc

Tham khảo các mục tiêu LPI

[LPIC-1 version 5.0, Exam 102, Objective 107.2](#)

Khối lượng

4

Các lĩnh vực kiến thức chính

- Quản lý các công việc cron và at.
- Định cấu hình quyền truy cập của người dùng vào cron và tại các dịch vụ.
- Hiểu về các đơn vị bộ hẹn giờ của systemd.

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- `/etc/cron.{d,daily,hourly,monthly,weekly}/`
- `/etc/at.deny`
- `/etc/at.allow`
- `/etc/crontab`
- `/etc/cron.allow`
- `/etc/cron.deny`
- `/var/spool/cron/`
- `crontab`
- `at`
- `atq`

- `atrm`
- `systemctl`
- `systemd-run`



107.2 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	107 Các Tác vụ Quản trị
Mục tiêu:	107.2 Tự động hóa các Tác vụ Quản trị Hệ thống bằng cách lập lịch trình công việc
Bài:	1 trên 2

Giới thiệu

Một trong những nhiệm vụ quan trọng nhất của một quản trị viên ưu tú là lập lịch trình cho các công việc cần được thực thi một cách thường xuyên. Ví dụ: quản trị viên có thể tạo và tự động hóa các công việc sao lưu, nâng cấp hệ thống và thực hiện nhiều hoạt động lặp đi lặp lại khác. Để làm được điều này, chúng ta có thể sử dụng tiện ích `cron`. Tiện ích này rất hữu ích trong việc tự động hóa tác vụ lập lịch trình.

Lên lịch công việc với Cron

Trong Linux, `cron` là một trình nền (daemon) chạy liên tục và sẽ "thức dậy" mỗi phút để kiểm tra một tập hợp các bảng nhằm tìm ra các tác vụ cần được thực thi. Các bảng này được gọi là *bảng công việc định kỳ* (crontabs) và chúng sẽ chứa các *công việc định kỳ* (cron jobs). Cron phù hợp với các máy chủ và hệ thống được bật nguồn liên tục vì mỗi công việc định kỳ sẽ chỉ được thực thi nếu hệ thống chạy đúng theo thời gian đã định. Nó có thể được sử dụng bởi người dùng thông thường (mỗi người dùng sẽ có *bảng công việc định kỳ* của riêng mình) cũng như siêu người dùng quản lý các bảng công việc định kỳ hệ thống.

NOTE

Trong Linux còn có tiện ích `anacron` phù hợp với các hệ thống có thể cần tắt nguồn (chẳng hạn như máy tính hoặc máy tính xách tay). Nó chỉ có thể được sử dụng bởi siêu người dùng. Nếu máy tắt khi các công việc `anacron` cần phải được thực thi, chúng sẽ chạy ngay khi máy bật. `anacron` nằm ngoài phạm vi của chứng chỉ LPIC-1.

Bảng công việc định kỳ của Người dùng

Bảng công việc định kỳ của người dùng là các tệp văn bản quản lý việc lên lịch cho các công việc định kỳ do người dùng chỉ định. Các tệp này luôn được đặt tên theo tài khoản của người dùng đã tạo ra chúng, nhưng vị trí của tệp sẽ phụ thuộc vào bản phân phối được sử dụng (thường là một thư mục con của `/var/spool/cron`).

Mỗi dòng trong một bảng công việc định kỳ của người dùng sẽ chứa sáu trường được phân tách bằng dấu cách:

- Phút trong giờ (0-59).
- Giờ trong ngày (0-23).
- Ngày trong tháng (1-31).
- Tháng trong năm (1-12).
- Ngày trong tuần (0-7, trong đó, Chủ nhật=0 hoặc Chủ nhật=7).
- Lệnh sẽ chạy.

Đối với tháng trong năm và ngày trong tuần, chúng ta có thể sử dụng ba chữ cái đầu tiên của tên thay vì số tương ứng.

Năm trường đầu tiên cho biết thời điểm thực thi lệnh được chỉ định trong trường thứ sáu và chúng có thể chứa một hoặc nhiều giá trị. Chúng ta có thể chỉ định nhiều giá trị bằng cách sử dụng:

*** (dấu hoa thị)**

Đề cập đến bất kỳ một giá trị nào.

, (dấu phẩy)

Chỉ định một danh sách các giá trị khả thi.

- (dấu gạch ngang)

Chỉ định một phạm vi các giá trị khả thi.

/ (dấu gạch chéo)

Chỉ định khoảng thời gian nghỉ.

Nhiều bản phân phối sẽ có cả tệp `/etc/crontab` có thể được sử dụng làm tài liệu tham khảo cho bố cục của tệp `cron`. Dưới đây là một tệp `/etc/crontab` ví dụ từ một bản cài đặt Debian:

```

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# Example of job definition:
# .----- minute (0 - 59)
# | .----- hour (0 - 23)
# | | .----- day of month (1 - 31)
# | | | .----- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# | | | | |
# * * * * * user-name command to be executed

```

Bảng công việc định kỳ của Hệ thống

Bảng công việc định kỳ của hệ thống là các tệp văn bản quản lý việc lên lịch cho các công việc định kỳ của hệ thống mà chỉ siêu người dùng mới có thể chỉnh sửa được. `/etc/crontab` và tất cả các tệp trong thư mục `/etc/cron.d` đều là các bảng công việc định kỳ hệ thống.

Hầu hết các bản phân phối đều có các thư mục `/etc/cron.hourly`, `/etc/cron.daily`, `/etc/cron.weekly` và `/etc/cron.monthly` chứa các tệp lệnh để chạy với tần suất thích hợp. Ví dụ: nếu muốn chạy tệp lệnh đó hàng ngày, ta có thể đặt tệp lệnh đó vào `/etc/cron.daily`.

WARNING

Một số bản phân phối sẽ sử dụng `/etc/cron.d/hourly`, `/etc/cron.d/daily`, `/etc/cron.d/weekly` và `/etc/cron.d/monthly`. Hãy luôn nhớ kiểm tra các thư mục chính xác để đặt các tệp lệnh mà bạn muốn `cron` chạy.

Cú pháp của bảng công việc định kỳ hệ thống cũng tương tự như cú pháp của bảng công việc định kỳ người dùng. Tuy nhiên, nó sẽ yêu cầu một trường bổ sung bắt buộc để chỉ định người dùng nào sẽ chạy một công việc định kỳ nhất định. Do đó, mỗi dòng trong bảng công việc định kỳ hệ thống sẽ chứa bảy trường được phân tách bằng dấu cách:

- Phút trong giờ (0-59).
- Giờ trong ngày (0-23).

- Ngày trong tháng (1-31).
- Tháng trong năm (1-12).
- Ngày trong tuần (0-7, trong đó, Chủ nhật=0 hoặc Chủ nhật=7).
- Tên tài khoản người dùng sẽ được sử dụng khi thực hiện lệnh.
- Lệnh sẽ chạy.

Đối với bảng công việc định kỳ của người dùng, chúng ta có thể chỉ định nhiều giá trị cho các trường thời gian bằng cách sử dụng các toán tử *, , - và /. Ta cũng có thể chỉ ra tháng trong năm và ngày trong tuần bằng ba chữ cái đầu tiên của tên thay vì số tương ứng.

Thông số thời gian cụ thể

Khi chỉnh sửa tệp bảng công việc định kỳ, chúng ta cũng có thể sử dụng các phím tắt đặc biệt trong năm cột đầu tiên thay vì thông số thời gian:

@reboot

Chạy tác vụ được chỉ định một lần sau khi khởi động lại.

@hourly

Chạy tác vụ được chỉ định mỗi giờ một lần vào thời điểm bắt đầu của giờ.

@daily (hoặc @midnight)

Chạy tác vụ được chỉ định mỗi ngày một lần vào lúc nửa đêm.

@weekly

Chạy tác vụ được chỉ định mỗi tuần một lần vào nửa đêm Chủ nhật.

@monthly

Chạy tác vụ được chỉ định mỗi tháng một lần vào lúc nửa đêm của ngày đầu tiên của tháng.

@yearly (hoặc @annual)

Chạy tác vụ được chỉ định mỗi năm một lần vào nửa đêm ngày 1 tháng 1.

Biến Bảng công việc định kỳ

Trong tệp bảng công việc định kỳ đôi khi sẽ có các phép gán biến được xác định trước khi khai báo các tác vụ theo lịch trình. Các biến môi trường thường được đặt là:

HOME

Thư mục nơi cron gọi các lệnh (theo mặc định là thư mục chính của người dùng).

MAILTO

Tên của người dùng hoặc địa chỉ mà đầu ra tiêu chuẩn và lỗi được gửi đến (theo mặc định là chủ sở hữu bảng công việc định kỳ). Người dùng có thể sử dụng nhiều giá trị được phân tách bằng dấu phẩy và một giá trị trống cho biết rằng không cần gửi bất kỳ một thư nào.

PATH

Đường dẫn có thể tìm thấy các lệnh.

SHELL

Vỏ sẽ sử dụng (theo mặc định là /bin/sh).

Tạo công việc định kỳ của Người dùng

Lệnh `crontab` được sử dụng để duy trì các tệp bảng công việc định kỳ cho từng người dùng. Cụ thể, ta có thể chạy lệnh `crontab -e` để chỉnh sửa tệp bảng công việc định kỳ của riêng mình hoặc tạo một bảng mới nếu nó chưa tồn tại.

```
$ crontab -e
no crontab for frank - using an empty one

Select an editor. To change later, run 'select-editor'.
 1. /bin/ed
 2. /bin/nano      < ---- easiest
 3. /usr/bin/emacs24
 4. /usr/bin/vim.tiny

Choose 1-4 [2]:
```

Theo mặc định, lệnh `crontab` sẽ mở trình chỉnh sửa được chỉ định bởi các biến môi trường `VISUAL` hoặc `EDITOR` để người dùng có thể bắt đầu chỉnh sửa tệp bảng công việc định kỳ của mình bằng trình chỉnh sửa ưa thích. Một số bản phân phối (như trong ví dụ trên) sẽ cho phép chúng ta chọn trình chỉnh sửa từ danh sách khi `crontab` được chạy lần đầu tiên.

Nếu muốn chạy tệp lệnh `foo.sh` nằm trong thư mục chính của mình vào lúc 10:00 sáng hàng ngày, chúng ta có thể thêm dòng sau vào tệp bảng công việc định kỳ:

```
0 10 * * * /home/frank/foo.sh
```

Hãy xem xét các mục bảng công việc định kỳ mẫu sau:

```
0,15,30,45 08 * * 2 /home/frank/bar.sh
30 20 1-15 1,6 1-5 /home/frank/foobar.sh
```

Dòng đầu tiên cho biết tệp lệnh `bar.sh` sẽ được thực thi vào Thứ Ba hàng tuần lúc 08:00 sáng, 08:15 sáng, 08:30 sáng và 08:45 sáng. Dòng thứ hai cho biết tệp lệnh `foobar.sh` sẽ được thực thi lúc 08:30 tối từ thứ Hai đến thứ Sáu trong mười lăm ngày đầu tiên của tháng Một và tháng Sáu.

WARNING

Mặc dù các tệp bảng công việc định kỳ có thể được chỉnh sửa thủ công nhưng người dùng vẫn được khuyến nghị sử dụng lệnh `crontab`. Các quyền trên tệp bảng công việc định kỳ thường chỉ được chỉnh sửa thông qua lệnh `crontab`.

Ngoài tùy chọn `-e` được đề cập ở trên, lệnh `crontab` còn có các tùy chọn hữu ích khác:

-l

Hiển thị bảng công việc định kỳ hiện tại trên đầu ra tiêu chuẩn.

-r

Xóa bảng công việc định kỳ hiện tại.

-u

Chỉ định tên của người dùng có bảng công việc định kỳ cần sửa đổi. Tùy chọn này yêu cầu quyền gốc và cho phép siêu người dùng chỉnh sửa các tệp bảng công việc định kỳ của người dùng.

Tạo công việc định kỳ cho Hệ thống

Không giống như bảng công việc định kỳ của người dùng, bảng công việc định kỳ hệ thống được cập nhật bằng một trình chỉnh sửa. Do đó, chúng ta không cần chạy lệnh `crontab` để chỉnh sửa `/etc/crontab` và các tệp trong `/etc/cron.d`. Hãy nhớ rằng khi chỉnh sửa các bảng công việc định kỳ hệ thống, ta phải chỉ định tài khoản sẽ được sử dụng để chạy công việc định kỳ đó (thường là siêu người dùng).

Ví dụ: nếu muốn chạy tệp lệnh `barfoo.sh` nằm trong thư mục `/root` hàng ngày lúc 01:30 sáng, ta có thể mở `/etc/crontab` bằng trình chỉnh sửa ưa thích của mình và thêm dòng sau:

```
30 01 * * * root /root/barfoo.sh >>/root/output.log 2>>/root/error.log
```

Trong ví dụ trên, đầu ra của công việc đã được thêm vào `/root/output.log`, trong khi các lỗi đã

được thêm vào `/root/error.log`.

WARNING

Trừ khi đầu ra được chuyển hướng đến một tệp như trong ví dụ trên (hoặc biến MAILTO được đặt thành giá trị trống), tất cả đầu ra từ một công việc định kỳ sẽ được gửi đến người dùng qua e-mail. Một thao tác phổ biến là chuyển hướng đầu ra tiêu chuẩn sang `/dev/null` (hoặc tới một tệp để xem lại sau nếu cần) và không chuyển hướng lỗi tiêu chuẩn. Bằng cách này, người dùng sẽ được thông báo ngay lập tức bằng e-mail về bất kỳ lỗi nào.

Định cấu hình Quyền Truy cập vào tác vụ lập lịch trình công việc

Trong Linux, các tệp `/etc/cron.allow` và `/etc/cron.deny` được sử dụng để thiết lập các hạn chế cho *bảng công việc định kỳ*. Đặc biệt, chúng được sử dụng để cho phép hoặc không cho phép lập lịch trình công việc định kỳ cho những người dùng khác nhau. Nếu `/etc/cron.allow` tồn tại, chỉ những người dùng (không phải siêu người dùng) được liệt kê trong đó mới có thể lập lịch trình các công việc định kỳ bằng lệnh `crontab`. Nếu `/etc/cron.allow` không tồn tại nhưng `/etc/cron.deny` có tồn tại, chỉ những người dùng (không phải siêu người dùng) được liệt kê trong tệp này mới không thể lập lịch trình các công việc định kỳ bằng lệnh `crontab` (trong trường hợp này là một tệp `/etc/cron.deny` trống, có nghĩa là tất cả mọi người dùng đều được phép lập lịch trình các công việc định kỳ bằng `crontab`). Nếu cả hai tệp này đều không tồn tại thì quyền truy cập của người dùng vào tác vụ lập lịch trình công việc định kỳ sẽ phụ thuộc vào bản phân phối được sử dụng.

NOTE

Các tệp `/etc/cron.allow` và `/etc/cron.deny` có chứa danh sách tên người dùng, mỗi tên sẽ nằm trên một dòng riêng biệt.

Một giải pháp thay thế cho Cron

Bằng cách sử dụng `systemd` làm trình quản lý hệ thống và dịch vụ, chúng ta có thể đặt các `_` bộ hẹn giờ (timer) thay thế cho `cron` để lên lịch cho các tác vụ của mình. Bộ hẹn giờ là các tệp đơn vị `systemd` được xác định bằng hậu tố `.timer`. Mỗi tệp này phải có một tệp đơn vị tương ứng mô tả đơn vị sẽ được kích hoạt khi đến giờ hẹn. Theo mặc định, một `timer` sẽ kích hoạt một dịch vụ có cùng tên (không tính hậu tố).

Một bộ hẹn giờ sẽ bao gồm một phần `[Timer]` chỉ định thời điểm để các công việc đã được lên lịch chạy. Cụ thể, ta có thể sử dụng tùy chọn `OnCalendar=` để xác định *bộ tính giờ thời gian thực* hoạt động theo cách tương tự như các công việc định kỳ (chúng dựa trên các biểu thức sự kiện lịch). Tùy chọn `OnCalendar=` yêu cầu cú pháp sau:

```
DayOfWeek Year-Month-Day Hour:Minute:Second
```

với `DayOfWeek` (ngày trong tuần) là tùy chọn. Các toán tử `*`, `/` và `,` có cùng ý nghĩa như các toán tử được sử dụng cho công việc định kỳ, trong khi chúng ta có thể sử dụng `..` giữa hai giá trị để biểu thị một phạm vi liên tiếp. Đối với đặc tả `DayOfWeek`, ta có thể sử dụng ba chữ cái đầu tiên của tên hoặc tên đầy đủ.

NOTE

Chúng ta cũng có thể xác định *bộ hẹn giờ đơn điệu* kích hoạt sau một khoảng thời gian trôi qua kể từ một điểm bắt đầu cụ thể (ví dụ: khi máy được khởi động hoặc khi chính bộ hẹn giờ được kích hoạt).

Ví dụ: nếu muốn chạy dịch vụ có tên `/etc/systemd/system/foobar.service` lúc 05:30 vào Thứ Hai đầu tiên của mỗi tháng, ta có thể thêm các dòng sau vào tệp đơn vị `/etc/systemd/system/foobar.timer` tương ứng.

```
[Unit]
Description=Run the foobar service

[Timer]
OnCalendar=Mon *-*-1..7 05:30:00
Persistent=true

[Install]
WantedBy=timers.target
```

Một khi đã tạo bộ hẹn giờ mới, chúng ta có thể kích hoạt và khởi động nó bằng cách chạy các lệnh sau với quyền gốc:

```
# systemctl enable foobar.timer
# systemctl start foobar.timer
```

Chúng ta có thể thay đổi tần suất công việc đã lên lịch của mình, sửa đổi giá trị `OnCalendar` rồi nhập lệnh `systemctl daemon-reload`.

Cuối cùng, nếu muốn xem danh sách các bộ hẹn giờ đang hoạt động được sắp xếp theo thời gian trôi qua tiếp theo, ta có thể sử dụng lệnh `systemctl list-timers`. Chúng ta cũng có thể thêm tùy chọn `--all` để xem các đơn vị hẹn giờ không hoạt động.

NOTE

Hãy nhớ rằng bộ hẹn giờ sẽ được ghi vào nhật ký `systemd` và bạn có thể xem lại nhật ký của các đơn vị khác nhau bằng lệnh `journalctl`. Cũng hãy nhớ rằng nếu đang hoạt động như một người dùng thông thường, bạn cần sử dụng tùy chọn `--user` của lệnh `systemctl` và `journalctl`.

Thay vì dạng chuẩn hóa chi tiết được đề cập ở trên, chúng ta có thể sử dụng một số biểu thức đặc biệt mô tả tần suất cụ thể để thực hiện công việc:

hourly

Chạy tác vụ được chỉ định mỗi giờ một lần vào thời điểm bắt đầu giờ.

daily

Chạy tác vụ được chỉ định mỗi ngày một lần vào lúc nửa đêm.

weekly

Chạy tác vụ được chỉ định mỗi tuần một lần vào nửa đêm thứ Hai.

monthly

Chạy tác vụ được chỉ định mỗi tháng một lần vào lúc nửa đêm của ngày đầu tiên của tháng.

yearly

Chạy tác vụ được chỉ định mỗi năm một lần vào nửa đêm ngày đầu tiên của tháng Một.

Hãy xem các trang hướng dẫn để biết danh sách đầy đủ về thông số kỹ thuật, ngày và giờ tại `systemd.timer(5)`.

Bài tập Hướng dẫn

1. Đối với mỗi phím tắt `crontab` sau đây, hãy cho biết thông số thời gian tương ứng (tức là năm cột đầu tiên trong tệp `crontab` của người dùng):

@hourly	
@daily	
@weekly	
@monthly	
@annually	

2. Đối với mỗi phím tắt `OnCalendar` sau đây, hãy chỉ ra thông số thời gian tương ứng (dạng chuẩn hóa chi tiết):

hourly	
daily	
weekly	
monthly	
yearly	

3. Hãy giải thích ý nghĩa của các thông số về thời gian sau đây được tìm thấy trong tệp `crontab`:

30 13 * * 1-5	
00 09-18 * * *	
30 08 1 1 *	
0,20,40 11 * * Sun	
00 09 10-20 1-3 *	
*/20 * * * *	

4. Hãy giải thích ý nghĩa của các thông số thời gian sau đây được sử dụng trong tùy chọn `OnCalendar` của một tệp bộ hẹn giờ:

--* 08:30:00	
Sat,Sun *-*-* 05:00:00	

- -01 13:15,30,45:00	
Fri *-09..12-* 16:20:00	
Mon,Tue *-*-1,15 08:30:00	
--* *:00/05:00	

Bài tập Mở rộng

1. Giả sử rằng bạn được phép lập lịch trình công việc bằng `cron` với tư cách là một người dùng thông thường, bạn sẽ sử dụng lệnh nào để tạo tệp bảng công việc định kỳ của riêng mình?

2. Hãy tạo một công việc được lên lịch đơn giản để thực thi lệnh `date` vào lúc 01:00 chiều Thứ Sáu hàng tuần. Bạn có thể thấy đầu ra của công việc này ở đâu?

3. Hãy tạo một công việc được lên lịch khác để thực thi tệp lệnh `foobar.sh` mỗi phút, chuyển hướng đầu ra đến tệp `output.log` trong thư mục chính để chỉ có lỗi tiêu chuẩn được gửi cho bạn qua e-mail.

4. Hãy xem mục `crontab` của công việc định kỳ mới được tạo. Tại sao không cần thiết phải chỉ định đường dẫn tuyệt đối của tệp lưu đầu ra tiêu chuẩn? Và tại sao bạn có thể sử dụng lệnh `./foobar.sh` để thực thi tệp lệnh?

5. Hãy chỉnh sửa mục nhập `crontab` trước đó bằng cách xóa chuyển hướng đầu ra và vô hiệu hoá công việc định kỳ đầu tiên bạn đã tạo.

6. Làm cách nào để có thể gửi kết quả và lỗi của công việc định kỳ tới tài khoản người dùng `emma` qua e-mail? Và làm thế nào để có thể tránh gửi đầu ra tiêu chuẩn và lỗi qua e-mail?

7. Hãy thực hiện lệnh `ls -l /usr/bin/crontab`. Bit đặc biệt nào sẽ được thiết lập và ý nghĩa của nó là gì?

Tóm tắt

Trong bài học này, chúng ta đã học về:

- Sử dụng `cron` để chạy các công việc định kỳ.
- Quản lý công việc định kỳ.
- Định cấu hình quyền truy cập của người dùng đối với tác vụ lập lịch trình công việc định kỳ.
- Hiểu vai trò của đơn vị hẹn giờ `systemd` thay thế cho `cron`.

Các tệp và lệnh sau đã được thảo luận trong bài học này:

`crontab`

Duy trì các tệp `crontab` cho người dùng cá nhân.

`/etc/cron.allow` và `/etc/cron.deny`

Các tệp cụ thể được sử dụng để thiết lập các hạn chế cho `crontab`.

`/etc/crontab`

Tệp bảng công việc định kỳ hệ thống.

`/etc/cron.d`

Thư mục chứa các tệp bảng công việc định kỳ hệ thống.

`systemctl`

Kiểm soát hệ thống `systemd` và quản lý dịch vụ. Liên quan đến bộ hẹn giờ, nó có thể được sử dụng để kích hoạt và khởi động chúng.

Đáp án Bài tập Hướng dẫn

1. Đối với mỗi phím tắt `crontab` sau đây, hãy cho biết thông số thời gian tương ứng (tức là năm cột đầu tiên trong tệp `crontab` của người dùng):

@hourly	0 * * * *
@daily	0 0 * * *
@weekly	0 0 * * 0
@monthly	0 0 1 * *
@annually	0 0 1 1 *

2. Đối với mỗi phím tắt `OnCalendar` sau đây, hãy chỉ ra thông số thời gian tương ứng (dạng chuẩn hóa chi tiết):

hourly	*-*-* *:00:00
daily	*-*-* 00:00:00
weekly	Mon *-*-* 00:00:00
monthly	*-*-01 00:00:00
yearly	*-01-01 00:00:00

1. Hãy giải thích ý nghĩa của các thông số về thời gian sau đây được tìm thấy trong tệp `crontab`:

30 13 * * 1-5	Vào lúc 01h30 chiều các ngày trong tuần từ thứ Hai đến thứ Sáu
00 09-18 * * *	Hàng ngày và hàng giờ từ 09 giờ sáng đến 06 giờ chiều
30 08 1 1 *	Vào lúc 08h30 của ngày đầu tiên của tháng 1
0,20,40 11 * * Mặt trời	Chủ nhật hàng tuần lúc 11 giờ sáng, 11 giờ 20 sáng và 11 giờ 40 sáng
00 09 10-20 1-3 *	Vào lúc 09h00 từ ngày 10 đến ngày 20 tháng 1, tháng 2 và tháng 3
*/20 * * * *	Cứ hai mươi phút một lần

2. Hãy giải thích ý nghĩa của các thông số thời gian sau đây được sử dụng trong tùy chọn `OnCalendar` của một tệp bộ hẹn giờ:

--* 08:30:00	Hàng ngày vào lúc 08:30 sáng
Sat,Sun *-*-* 05:00:00	Vào lúc 05h00 sáng thứ bảy và chủ nhật
--*01 13:15,30,45:00	Vào lúc 01h15, 01h30 và 01h45 chiều những ngày đầu tiên của tháng
Fri *-09..12-* 16:20:00	Vào lúc 04:20 chiều Thứ Sáu hàng tuần trong tháng 9, tháng 10, tháng 11 và tháng 12
Mon,Tue *-*-1,15 08:30:00	Vào lúc 08h30 sáng ngày đầu tiên hoặc ngày 15 hàng tháng nếu ngày đó là thứ Hai hoặc thứ Ba
--* *:00/05:00	Cứ năm phút một lần

Đáp án Bài tập Mở rộng

- Giả sử rằng bạn được phép lập lịch trình công việc bằng `cron` với tư cách là một người dùng thông thường, bạn sẽ sử dụng lệnh nào để tạo tệp bảng công việc định kỳ của riêng mình?

```
dave@hostname ~ $ crontab -e
no crontab for dave - using an empty one

Select an editor. To change later, run 'select-editor'.
 1. /bin/ed
 2. /bin/nano      < ---- easiest
 3. /usr/bin/emacs24
 4. /usr/bin/vim.tiny

Choose 1-4 [2]:
```

- Hãy tạo một công việc được lên lịch đơn giản để thực thi lệnh `date` vào lúc 01:00 chiều Thứ Sáu hàng tuần. Bạn có thể thấy đầu ra của công việc này ở đâu?

```
00 13 * * 5 date
```

Đầu ra đã được gửi tới người dùng; để xem nó, bạn có thể sử dụng lệnh `mail`.

- Hãy tạo một công việc được lên lịch khác để thực thi tệp lệnh `foobar.sh` mỗi phút, chuyển hướng đầu ra đến tệp `output.log` trong thư mục chính để chỉ có lỗi tiêu chuẩn được gửi cho bạn qua e-mail.

```
* /1 * * * * ./foobar.sh >> output.log
```

- Hãy xem mục `crontab` của công việc định kỳ mới được tạo. Tại sao không cần thiết phải chỉ định đường dẫn tuyệt đối của tệp lưu đầu ra tiêu chuẩn? Và tại sao bạn có thể sử dụng lệnh `./foobar.sh` để thực thi tệp lệnh?

`cron` sẽ gọi các lệnh từ thư mục chính của người dùng trừ khi một vị trí khác được chỉ định bởi biến môi trường `HOME` trong tệp `crontab`. Vì lý do này, bạn có thể sử dụng đường dẫn tương đối của tệp đầu ra và chạy tệp lệnh với `./foobar.sh`.

- Hãy chỉnh sửa mục nhập `crontab` trước đó bằng cách xóa chuyển hướng đầu ra và vô hiệu hoá công việc định kỳ đầu tiên bạn đã tạo.

```
#00 13 * * 5 date
*/1 * * * * ./foobar.sh
```

Để vô hiệu hoá một công việc định kỳ, bạn chỉ cần chú thích dòng tương ứng trong tệp `crontab`.

6. Làm cách nào để có thể gửi kết quả và lỗi của công việc định kỳ tới tài khoản người dùng `emma` qua e-mail? Và làm thế nào để có thể tránh gửi đầu ra tiêu chuẩn và lỗi qua e-mail?

Để gửi đầu ra tiêu chuẩn và lỗi tới `emma`, bạn phải đặt biến môi trường `MAILTO` trong tệp `crontab` của mình như sau:

```
MAILTO="emma"
```

Để báo cho cron rằng không cần gửi bất kỳ thư nào, bạn có thể gán một giá trị trống cho biến môi trường `MAILTO`.

```
MAILTO=""
```

7. Hãy thực hiện lệnh `ls -l /usr/bin/crontab`. Bit đặc biệt nào sẽ được thiết lập và ý nghĩa của nó là gì?

```
$ ls -l /usr/bin/crontab
-rwxr-sr-x 1 root crontab 25104 feb 10 2015 /usr/bin/crontab
```

Lệnh `crontab` có tập bit SGID (ký tự `s` thay vì `r` thực thi cho nhóm), có nghĩa là nó được thực thi với các đặc quyền của nhóm. Đây là lý do tại sao người dùng thông thường có thể chỉnh sửa tệp `crontab` của họ bằng lệnh `crontab`. Hãy lưu ý rằng nhiều bản phân phối có thể đặt quyền cho tệp để tệp `crontab` chỉ có thể được chỉnh sửa thông qua lệnh `crontab`.



107.2 Bài 2

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	107 Các Tác vụ Quản trị
Mục tiêu:	107.2 Tự động hóa các Tác vụ Quản trị Hệ thống bằng cách lập lịch trình công việc
Bài:	2 trên 2

Giới thiệu

Như đã học trong bài học trước, người dùng có thể lập lịch trình các công việc định kỳ bằng cách sử dụng `cron` hoặc bộ hẹn giờ `systemd`. Tuy nhiên, đôi khi chúng ta sẽ chỉ cần chạy một công việc một lần tại một thời điểm cụ thể trong tương lai. Để thực hiện việc này, ta có thể sử dụng một tiện ích mạnh mẽ khác là lệnh `at`.

Lập lịch trình công việc với `at`

Lệnh `at` được sử dụng để lập lịch trình cho các tác vụ chỉ cần thực thi một lần và sẽ chỉ yêu cầu chỉ định thời điểm được chạy trong tương lai. Sau khi nhập `at` trên dòng lệnh, theo sau là thông số thời gian, chúng ta sẽ nhập dấu nhắc `at` vào nơi các lệnh sẽ được thực thi. Chúng ta có thể thoát khỏi dấu nhắc bằng chuỗi phím `Ctrl + D`.

```
$ at now +5 minutes
warning: commands will be executed using /bin/sh
at> date
at> Ctrl+D
```



```
job 12 at Sat Sep 14 09:15:00 2019
```

Công việc của `at` trong ví dụ trên chỉ là thực thi lệnh `date` sau năm phút. Tương tự như `cron`, đầu ra tiêu chuẩn và lỗi sẽ được gửi cho người dùng qua e-mail. Hãy lưu ý rằng trình nền `atd` sẽ phải chạy trên hệ thống để có thể sử dụng lệnh lập lịch trình công việc `at`.

NOTE

Trong Linux, lệnh `batch` cũng tương tự như `at` nhưng `batch` sẽ chỉ được thực thi khi tải lượng hệ thống đủ thấp để cho phép nó chạy.

Các tùy chọn quan trọng nhất áp dụng cho lệnh `at` là:

-c

In các lệnh của một ID công việc cụ thể ra đầu ra tiêu chuẩn.

-d

Xóa công việc dựa trên ID công việc. Đây là bí danh của `atrm`.

-f

Đọc công việc từ một tệp thay vì đầu vào tiêu chuẩn.

-l

Liệt kê các công việc đang chờ xử lý của người dùng. Nếu là siêu người dùng, tất cả công việc của tất cả mọi người dùng sẽ được liệt kê. Đây là bí danh của `atq`.

-m

Gửi thư cho người dùng khi kết thúc công việc ngay cả khi không có đầu ra.

-q

Chỉ định một hàng đợi ở dạng một chữ cái từ `a` đến `z` và từ `A` đến `Z` (theo mặc định `a` cho `at` và `b` cho `batch`). Các công việc trong hàng đợi có chữ cái cao nhất được thực thi với độ chính xác cao hơn. Các công việc được gửi đến hàng đợi có chữ in hoa được coi là công việc của `batch`.

-v

Hiển thị thời gian công việc sẽ chạy trước khi đọc công việc.

Liệt kê các công việc đã được lập lịch trình với `atq`

Hãy cùng lên lịch cho hai công việc `at` nữa: công việc đầu tiên là thực thi tệp lệnh `foo.sh` lúc 09:30 sáng và công việc thứ hai là thực thi tệp lệnh `bar.sh` sau một giờ.

```
$ at 09:30 AM
warning: commands will be executed using /bin/sh
at> ./foo.sh
at> Ctrl+D
job 13 at Sat Sep 14 09:30:00 2019
$ at now +2 hours
warning: commands will be executed using /bin/sh
at> ./bar.sh
at> Ctrl+D
job 14 at Sat Sep 14 11:10:00 2019
```

Để liệt kê các công việc đang chờ được xử lý, chúng ta có thể sử dụng lệnh `atq` để hiển thị các thông tin sau cho từng công việc: ID công việc, ngày thực hiện công việc, thời gian thực hiện công việc, hàng đợi và tên người dùng.

```
$ atq
14      Sat Sep 14 11:10:00 2019 a frank
13      Sat Sep 14 09:30:00 2019 a frank
12      Sat Sep 14 09:15:00 2019 a frank
```

Hãy nhớ rằng lệnh `at -l` là bí danh của `atq`.

NOTE

Nếu chạy `atq` với quyền gốc, nó sẽ hiển thị các công việc được xếp hàng đợi cho tất cả người dùng.

Xóa công việc bằng `atrm`

Nếu muốn xóa một công việc `at`, chúng ta có thể sử dụng lệnh `atrm`, theo sau là ID công việc. Ví dụ: để xóa công việc có ID 14, ta có thể chạy lệnh sau:

```
$ atrm 14
```

Chúng ta có thể xóa nhiều công việc bằng `atrm` nếu chỉ định nhiều ID và phân tách chúng bằng dấu cách. Hãy nhớ rằng lệnh `at -d` chính là bí danh của `atrm`.

NOTE

Nếu chạy `atrm` với quyền gốc, bạn có thể xóa công việc của mọi người dùng.

Định cấu hình Quyền truy cập vào tác vụ lập lịch trình công việc

Việc ủy quyền cho người dùng thông thường lập lịch trình các công việc `at` được xác định bởi các tệp `/etc/at.allow` và `/etc/at.deny`. Nếu `/etc/at.allow` tồn tại, chỉ những người dùng không phải siêu người dùng được liệt kê trong đó mới có thể lập lịch trình công việc `at`. Nếu `/etc/at.allow` không tồn tại nhưng `/etc/at.deny` tồn tại, chỉ những người dùng không phải siêu người dùng được liệt kê trong đó mới không thể lên lịch các công việc `at` (trong trường hợp này là một tệp `/etc/at.deny` trống, tức là mọi người dùng đều được phép lên lịch các công việc `at`). Nếu cả hai tệp này đều không tồn tại thì quyền truy cập của người dùng vào tác vụ lập lịch trình công việc `at` sẽ phụ thuộc vào bản phân phối được sử dụng.

Thông số thời gian cụ thể

Chúng ta có thể chỉ định thời điểm thực hiện một công việc `at` cụ thể bằng cách sử dụng mẫu `GG:PP` (Giờ:Phút), theo sau là `AM` hoặc `PM` tùy ý trong trường hợp định dạng 12 giờ. Nếu thời gian chỉ định đã trôi qua, hệ thống sẽ tự hiểu rằng thời gian là vào ngày hôm sau. Nếu muốn lên lịch một ngày cụ thể mà công việc sẽ chạy, ta sẽ phải thêm thông tin ngày sau thời gian đó bằng một trong các mẫu sau: `tên-tháng ngày-trong-tháng`, `tên-tháng ngày-trong-tháng năm`, `MMDDYY`, `MM/DD/YY`, `DD.MM.YY` và `YYYY-MM-DD`).

Các từ khóa sau cũng sẽ được chấp nhận: `midnight`, `noon`, `teatime` (4 giờ chiều) và `now`, theo sau là dấu cộng (+) và khoảng thời gian (số phút, giờ, ngày và tuần). Cuối cùng, ta có thể yêu cầu `at` thực hiện công việc của hôm nay hoặc ngày mai bằng cách thêm hậu tố thời gian với các từ `today` hoặc `tomorrow`. Ví dụ: ta có thể sử dụng `at 07:15 AM Jan 01` (lúc 07:15 sáng ngày 01 tháng 1) để thực hiện công việc lúc 07:15 sáng ngày 01 tháng 1 và `at now +5 minutes` (hiện tại +5 phút) để thực hiện công việc trong 5 phút kể từ thời điểm hiện tại. Chúng ta có thể đọc tệp `timespec` trong cây thư mục `/usr/share` để biết thêm thông tin định nghĩa chính xác về thông số thời gian.

Một giải pháp thay thế cho `at`

Bằng việc sử dụng `systemd` làm trình quản lý hệ thống và dịch vụ, chúng ta cũng có thể lên lịch các tác vụ chạy một lần bằng lệnh `systemd-run`. Lệnh này thường được sử dụng để tạo một bộ hẹn giờ nhất thời để lệnh được thực thi tại một thời điểm cụ thể mà không cần phải tạo tệp dịch vụ. Ví dụ: với quyền gốc, người dùng có thể chạy lệnh `date` lúc 11:30 sáng ngày 10/10/2019 bằng cách sử dụng lệnh sau:

```
# systemd-run --on-calendar='2019-10-06 11:30' date
```

Nếu muốn chạy tệp lệnh `foo.sh` nằm trong thư mục làm việc hiện tại, sau hai phút, chúng ta có

thể sử dụng:

```
# systemd-run --on-active="2m" ./foo.sh
```

Hãy tham khảo các trang hướng dẫn để tìm hiểu tất cả các cách sử dụng của `systemd-run` với `systemd-run(1)`.

NOTE

Hãy nhớ rằng bộ hẹn giờ sẽ được ghi vào nhật ký `systemd` và bạn có thể xem lại nhật ký của các đơn vị khác nhau bằng lệnh `journalctl`. Cũng hãy nhớ rằng nếu đang hoạt động như một người dùng thông thường, bạn cần sử dụng tùy chọn `--user` của lệnh `systemctl` và `journalctl`.

Bài tập Hướng dẫn

1. Đối với mỗi thông số thời gian sau đây, hãy cho biết thông số nào hợp lệ và thông số nào không hợp lệ đối với `at`:

<code>at 08:30 AM next week</code>	
<code>at midday</code>	
<code>at 01-01-2020 07:30 PM</code>	
<code>at 21:50 01.01.20</code>	
<code>at now +4 days</code>	
<code>at 10:15 PM 31/03/2021</code>	
<code>at tomorrow 08:30 AM</code>	

2. Một khi đã lập lịch trình công việc với `at`, làm thế nào để có thể xem lại các lệnh của nó?

3. Bạn có thể sử dụng lệnh nào để xem lại các công việc `at` đang chờ xử lý? Bạn sẽ sử dụng lệnh nào để xóa chúng?

4. Với `systemd`, lệnh nào có thể được dùng thay thế cho `at`?

Bài tập Mở rộng

1. Giả sử bạn đang hoạt động như một người dùng thông thường, hãy tạo một công việc `at` để chạy tệp lệnh `foo.sh` nằm trong thư mục chính của bạn vào lúc 10:30 sáng ngày 31 tháng 10 sắp tới.

2. Hãy đăng nhập vào hệ thống với tư cách là một người dùng thông thường khác và tạo một công việc `at` khác để chạy tệp lệnh `bar.sh` vào lúc 10:00 sáng ngày mai (giả sử tệp lệnh nằm trong thư mục chính của người dùng).

3. Hãy đăng nhập vào hệ thống với tư cách là một người dùng thông thường khác và tạo một công việc `at` khác để chạy tệp lệnh `foobar.sh` chỉ sau 30 phút (giả sử tệp lệnh nằm trong thư mục chính của người dùng).

4. Bây giờ, với quyền gốc, hãy chạy lệnh `atq` để xem lại các công việc mà `at` đã lập lịch trình của tất cả mọi người dùng. Điều gì sẽ xảy ra nếu người dùng thông thường thực hiện lệnh này?

5. Với quyền gốc, hãy xóa tất cả các công việc `at` đang chờ được xử lý này bằng một lệnh duy nhất.

6. Hãy chạy lệnh `ls -l /usr/bin/at` và kiểm tra các quyền của nó.

Tóm tắt

Trong bài học này, chúng ta đã học về:

- Sử dụng `at` để chạy các công việc chạy một lần tại một thời điểm cụ thể.
- Quản lý các công việc `at`.
- Định cấu hình quyền truy cập của người dùng vào việc lập lịch trình công việc `at`.
- Sử dụng `systemd-run` thay thế cho `at`.

Các tệp và lệnh sau đã được thảo luận trong bài học này:

`at`

Thực hiện các lệnh tại một thời điểm nhất định.

`atq`

Liệt kê các công việc `at` đang chờ xử lý của người dùng (trừ khi người dùng là siêu người dùng).

`atrm`

Xóa các công việc `at` bằng mã số công việc của chúng.

`/etc/at.allow` và `/etc/at.deny`

Các tệp cụ thể được sử dụng để đặt giới hạn `at`.

`systemd-run`

Tạo và khởi động một đơn vị `timer` nhất thời thay thế cho `at` để lập lịch trình chạy một lần.

Đáp án Bài tập Hướng dẫn

1. Đối với mỗi thông số thời gian sau đây, hãy cho biết thông số nào hợp lệ và thông số nào không hợp lệ đối với `at`:

<code>at 08:30 AM next week</code>	Hợp lệ
<code>at midday</code>	Không hợp lệ
<code>at 01-01-2020 07:30 PM</code>	Không hợp lệ
<code>at 21:50 01.01.20</code>	Hợp lệ
<code>at now +4 days</code>	Hợp lệ
<code>at 10:15 PM 31/03/2021</code>	Không hợp lệ
<code>at tomorrow 08:30 AM monotonic</code>	Không hợp lệ

2. Một khi đã lập lịch trình công việc với `at`, làm thế nào để có thể xem lại các lệnh của nó?

Bạn có thể sử dụng lệnh `at -c`, theo sau là ID của công việc có lệnh bạn muốn xem lại. Hãy lưu ý rằng đầu ra cũng sẽ chứa hầu hết phần môi trường đang hoạt động tại thời điểm công việc được lên lịch. Hãy nhớ rằng siêu người dùng có thể xem lại công việc của mọi người dùng.

3. Bạn có thể sử dụng lệnh nào để xem lại các công việc `at` đang chờ xử lý? Bạn sẽ sử dụng lệnh nào để xóa chúng?

Bạn có thể sử dụng lệnh `at -l` để xem lại các công việc đang chờ được xử lý và lệnh `at -d` để xóa công việc của mình. `at -l` là bí danh của `atq` và `at -d` là bí danh của `atrm`. Hãy nhớ rằng siêu người dùng có thể liệt kê và xóa công việc của tất cả mọi người dùng.

4. Với `systemd`, lệnh nào có thể được dùng thay thế cho `at`?

Lệnh `systemd-run` có thể được sử dụng thay thế cho `at` để lên lịch các công việc chạy một lần. Ví dụ: bạn có thể sử dụng nó để chạy các lệnh tại một thời điểm cụ thể, xác định *bộ hẹn giờ lịch* hoặc *bộ hẹn giờ đơn điệu* liên quan đến các điểm bắt đầu khác nhau.

Đáp án Bài tập Mở rộng

- Giả sử bạn đang hoạt động như một người dùng thông thường, hãy tạo một công việc at để chạy tệp lệnh `foo.sh` nằm trong thư mục chính của bạn vào lúc 10:30 sáng ngày 31 tháng 10 sắp tới.

```
$ at 10:30 AM October 31
warning: commands will be executed using /bin/sh
at> ./foo.sh
at> Ctrl+D
job 50 at Thu Oct 31 10:30:00 2019
```

- Hãy đăng nhập vào hệ thống với tư cách là một người dùng thông thường khác và tạo một công việc at khác để chạy tệp lệnh `bar.sh` vào lúc 10:00 sáng ngày mai (giả sử tệp lệnh nằm trong thư mục chính của người dùng).

```
$ at 10:00 AM tomorrow
warning: commands will be executed using /bin/sh
at> ./bar.sh
at> Ctrl+D
job 51 at Sun Oct 6 10:00:00 2019
```

- Hãy đăng nhập vào hệ thống với tư cách là một người dùng thông thường khác và tạo một công việc at khác để chạy tệp lệnh `bar.sh` vào lúc 10:00 sáng ngày mai (giả sử tệp lệnh nằm trong thư mục chính của người dùng).

```
$ at now +30 minutes
warning: commands will be executed using /bin/sh
at> ./foobar.sh
at> Ctrl+D
job 52 at Sat Oct 5 10:19:00 2019
```

- Bây giờ, với quyền gốc, hãy chạy lệnh `atq` để xem lại các công việc mà at đã lập lịch trình của tất cả mọi người dùng. Điều gì sẽ xảy ra nếu người dùng thông thường thực hiện lệnh này?

```
# atq
52      Sat Oct 5 10:19:00 2019 a dave
50      Thu Oct 31 10:30:00 2019 a frank
51      Sun Oct 6 10:00:00 2019 a emma
```

Nếu chạy lệnh `atq` với quyền gốc, tất cả các công việc `at` đang chờ được xử lý của tất cả mọi người dùng sẽ được liệt kê. Nếu chạy nó như một người dùng thông thường thì chỉ các công việc `at` đang chờ được xử lý của người dùng đó mới được liệt kê.

- Với quyền gốc, hãy xóa tất cả các công việc `at` đang chờ được xử lý này bằng một lệnh duy nhất.

```
# atrm 50 51 52
```

- Hãy thực hiện lệnh `ls -l /usr/bin/crontab`. Bit đặc biệt nào sẽ được thiết lập và ý nghĩa của nó là gì?

```
# ls -l /usr/bin/at
-rwsr-sr-x 1 daemon daemon 43762 Dec  1  2015 /usr/bin/at
```

Trong bản phân phối này, lệnh `at` có cả tập bit SUID (ký tự `s` thay vì `r` thực thi cho chủ sở hữu) và SGID (ký tự `s` thay vì `r` thực thi cho nhóm), có nghĩa là nó được thực thi với các đặc quyền của chủ sở hữu và nhóm của tệp (`daemon` cho cả hai). Đây là lý do tại sao người dùng thông thường có thể lập lịch trình công việc bằng `at`.



107.3 Bản địa hoá và Quốc tế hoá

Tham khảo các mục tiêu LPI

[LPIC-1 version 5.0, Exam 102, Objective 107.3](#)

Khối lượng

3

Các lĩnh vực kiến thức chính

- Định cấu hình cài đặt ngôn ngữ và biến môi trường.
- Định cấu hình cài đặt múi giờ và biến môi trường.

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- `/etc/timezone`
- `/etc/localtime`
- `/usr/share/zoneinfo/`
- `LC_*`
- `LC_ALL`
- `LANG`
- `TZ`
- `/usr/bin/locale`
- `tzselect`
- `timedatectl`
- `date`
- `iconv`

- UTF-8
- ISO-8859
- ASCII
- Unicode



107.3 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	107 Các Tác vụ Quản trị
Mục tiêu:	107.3 Địa phương hoá và Quốc tế hoá
Bài:	1 trên 1

Giới thiệu

Tất cả các bản phân phối Linux lớn đều có thể được định cấu hình để sử dụng các cài đặt địa phương hóa tùy chỉnh. Các cài đặt này bao gồm các khái niệm về vùng và ngôn ngữ như múi giờ, ngôn ngữ giao diện cũng như mã hóa ký tự có thể được sửa đổi trong quá trình cài đặt hệ điều hành hoặc bất kỳ thời điểm nào sau này.

Các ứng dụng sẽ dựa vào các biến môi trường, tệp cấu hình hệ thống và lệnh để quyết định sử dụng thời gian và ngôn ngữ nào cho phù hợp. Do đó, hầu hết các bản phân phối Linux đều sử dụng chung một cách được tiêu chuẩn hóa để điều chỉnh cài đặt thời gian và địa phương hóa. Những điều chỉnh này rất quan trọng không chỉ trong việc cải thiện trải nghiệm người dùng mà còn để đảm bảo rằng thời gian của các sự kiện hệ thống — chẳng hạn như các sự kiện quan trọng cần báo cáo các vấn đề liên quan đến bảo mật — sẽ được tính toán một cách chính xác.

Để có thể biểu diễn bất kỳ một văn bản thuộc bất kể một ngôn ngữ nói nào, các hệ điều hành hiện đại cần có một *tiêu chuẩn mã hóa ký tự* tham chiếu và các hệ thống Linux cũng không phải là ngoại lệ. Vì máy tính chỉ có thể xử lý các con số nên mỗi ký tự văn bản đều sẽ không khác gì một con số gắn liền với một ký hiệu đồ họa. Các nền tảng máy tính biệt lập có thể sẽ gán các giá trị số khác nhau với cùng một ký tự, vì thế nên cần có một tiêu chuẩn mã hóa ký tự chung để khiến

chúng có khả năng tương thích. Một tài liệu văn bản được tạo trên một hệ thống sẽ chỉ có thể đọc được trên một hệ thống khác nếu cả hai cùng thống nhất về định dạng mã hóa và số hiệu liên kết với các ký tự, hoặc ít nhất là biết cách chuyển đổi giữa hai tiêu chuẩn.

Bản chất không đồng nhất của các cài đặt địa phương hóa trong hệ thống dựa trên Linux dẫn đến một số điểm khác biệt nhỏ giữa các bản phân phối. Dù vậy, tất cả các bản phân phối đều có chung các công cụ và khái niệm cơ bản để thiết lập các khía cạnh quốc tế hóa của một hệ thống.

Múi giờ

Múi giờ (Time zones) là các dải giờ riêng biệt có tỷ lệ gần đúng trên bề mặt Trái đất trải dài tương đương với một giờ, hay nói cách khác là các vùng trên thế giới sẽ trải qua cùng một giờ trong ngày tại bất kỳ thời điểm nào. Vì không có một kinh độ nào có thể được coi là điểm bắt đầu của ngày mới cho toàn thế giới nên các múi giờ đều sẽ tương quan với *kinh tuyến gốc* nơi kinh độ của Trái Đất có số đo góc được xác định là 0. Thời gian tại kinh tuyến gốc được gọi là *Giờ Phối hợp Quốc tế*, theo quy ước viết tắt là UTC. Vì lý do thực tế, các múi giờ sẽ không tuân theo khoảng cách dọc chính xác từ điểm tham chiếu (kinh tuyến gốc). Thay vào đó, chúng được điều chỉnh thủ công để tuân theo biên giới của các quốc gia hoặc các phân khu quan trọng khác.

Các phân khu chính trị quan trọng tới mức các múi giờ đã được đặt theo tên của một số tác nhân địa lý chính trong các khu vực cụ thể đó, thường là dựa trên tên của một quốc gia hoặc một thành phố lớn bên trong khu vực đó. Tuy vậy, các múi giờ vẫn được chia theo phần bù thời gian của chúng so với UTC và phần bù này cũng có thể được sử dụng để biểu thị vùng được đề cập. Ví dụ: múi giờ *GMT-5* cho biết thời gian UTC sớm hơn thời gian của khu vực này 5 giờ, hay nói cách khác là khu vực đó chậm hơn UTC 5 giờ. Tương tự, múi giờ *GMT+3* ám chỉ thời gian UTC chậm hơn thời gian của khu vực này ba giờ. Thuật ngữ GMT—từ *Greenwich Mean Time* (Giờ chuẩn Greenwich)—được sử dụng như từ đồng nghĩa với UTC trong tên vùng dựa trên phần bù.

Một máy được kết nối có thể được truy cập từ nhiều nơi khác nhau trên thế giới. Do đó, cách tốt nhất là đặt đồng hồ phần cứng thành UTC (múi giờ GMT+0) và dành việc lựa chọn múi giờ cho từng trường hợp cụ thể. Ví dụ: các dịch vụ đám mây thường được định cấu hình để sử dụng UTC vì nó có thể giúp giảm thiểu sự không nhất quán thường xuyên giữa giờ cục bộ và thời gian tại máy khách hoặc tại các máy chủ khác. Ngược lại, người dùng mở phiên từ xa trên máy chủ có thể sẽ muốn sử dụng múi giờ cục bộ của họ. Như vậy, tùy từng trường hợp mà hệ điều hành sẽ thiết lập múi giờ chính xác sao cho phù hợp.

Ngoài ngày giờ hiện tại, lệnh `date` cũng sẽ in múi giờ hiện được cấu hình:

```
$ date
```

```
Mon Oct 21 10:45:21 -03 2019
```

Giá trị chênh lệch so với UTC được thể hiện bởi giá trị `-03`, nghĩa là thời gian được hiển thị ít hơn UTC ba giờ. Do đó, thời gian UTC sớm hơn ba giờ và khiến cho `GMT-3` trở thành múi giờ tương ứng cho thời điểm sử dụng. Lệnh `timedatectl` có sẵn trong các bản phân phối sử dụng `systemd` sẽ hiển thị thêm chi tiết về ngày giờ của hệ thống:

```
$ timedatectl
           Local time: Sat 2019-10-19 17:53:18 -03
           Universal time: Sat 2019-10-19 20:53:18 UTC
           RTC time: Sat 2019-10-19 20:53:18
           Time zone: America/Sao_Paulo (-03, -0300)
System clock synchronized: yes
systemd-timesyncd.service active: yes
           RTC in local TZ: no
```

Như được hiển thị trong mục `Time zone`, tên múi giờ được dựa trên địa phương—như `America/Sao_Paulo`—cũng sẽ được chấp nhận. Múi giờ mặc định cho hệ thống được lưu trong tệp `/etc/timezone`, có thể là bằng tên mô tả đầy đủ hoặc là phần bù của giờ khu vực. Tên múi giờ chung do chênh lệch UTC đưa ra phải có `Etc` đứng đầu tên. Vì vậy, để đặt múi giờ mặc định thành `GMT+3`, tên của múi giờ phải là `Etc/GMT+3`:

```
$ cat /etc/timezone
Etc/GMT+3
```

Mặc dù tên múi giờ dựa trên địa phương không yêu cầu phần bù thời gian để hoạt động nhưng để lựa chọn chúng cũng không phải là một việc dễ dàng. Cùng một khu vực có thể có nhiều tên nên rất khó nhớ. Để giải quyết vấn đề này, lệnh `tzselect` có cung cấp một phương thức tương tác và sẽ hướng dẫn người dùng định nghĩa múi giờ một cách chính xác. Lệnh `tzselect` phải có sẵn theo mặc định trong tất cả các bản phân phối Linux vì nó được cung cấp bởi gói chứa các chương trình tiện ích cần thiết liên quan đến Thư viện GNU C.

Ví dụ: lệnh `tzselect` sẽ hữu ích đối với người dùng muốn xác định múi giờ cho “São Paulo City” ở “Brazil”. `tzselect` sẽ bắt đầu bằng cách hỏi vùng vĩ mô của vị trí mong muốn:

```
$ tzselect
Please identify a location so that time zone rules can be set correctly.
Please select a continent, ocean, "coord", or "TZ".
 1) Africa
 2) Americas
 3) Antarctica
 4) Asia
```

```

5) Atlantic Ocean
6) Australia
7) Europe
8) Indian Ocean
9) Pacific Ocean
10) coord - I want to use geographical coordinates.
11) TZ - I want to specify the time zone using the Posix TZ format.
#? 2

```

Tùy chọn 2 dành cho các địa điểm (Bắc và Nam) của Châu Mỹ và không nhất thiết phải ở cùng múi giờ. Ta cũng có thể chỉ định múi giờ bằng tọa độ địa lý hoặc bằng ký hiệu phần bù hay còn được gọi là định dạng *Posix TZ*. Bước tiếp theo là chọn quốc gia:

```

Please select a country whose clocks agree with yours.
1) Anguilla          19) Dominican Republic  37) Peru
2) Antigua & Barbuda 20) Ecuador              38) Puerto Rico
3) Argentina        21) El Salvador         39) St Barthelemy
4) Aruba            22) French Guiana       40) St Kitts & Nevis
5) Bahamas         23) Greenland           41) St Lucia
6) Barbados        24) Grenada             42) St Maarten (Dutch)
7) Belize          25) Guadeloupe         43) St Martin (French)
8) Bolivia         26) Guatemala          44) St Pierre & Miquelon
9) Brazil          27) Guyana              45) St Vincent
10) Canada         28) Haiti               46) Suriname
11) Caribbean NL  29) Honduras           47) Trinidad & Tobago
12) Cayman Islands 30) Jamaica             48) Turks & Caicos Is
13) Chile          31) Martinique         49) United States
14) Colombia      32) Mexico              50) Uruguay
15) Costa Rica    33) Montserrat         51) Venezuela
16) Cuba          34) Nicaragua          52) Virgin Islands (UK)
17) Curaçao       35) Panama              53) Virgin Islands (US)
18) Dominica     36) Paraguay
#? 9

```

Lãnh thổ của Brazil trải dài bốn múi giờ nên chỉ thông tin quốc gia là không đủ để đặt múi giờ. Trong bước tiếp theo, `tzselect` sẽ yêu cầu người dùng chỉ định vùng cục bộ:

```

Please select one of the following time zone regions.
1) Atlantic islands
2) Pará (east); Amapá
3) Brazil (northeast: MA, PI, CE, RN, PB)
4) Pernambuco

```



```

5) Tocantins
6) Alagoas, Sergipe
7) Bahia
8) Brazil (southeast: GO, DF, MG, ES, RJ, SP, PR, SC, RS)
9) Mato Grosso do Sul
10) Mato Grosso
11) Pará (west)
12) Rondônia
13) Roraima
14) Amazonas (east)
15) Amazonas (west)
16) Acre
#? 8

```

Không phải tất cả các tên địa phương đều có sẵn nhưng chúng ta cũng chỉ cần chọn khu vực gần nhất đã là đủ. Thông tin đã cho sau đó sẽ được `tzselect` sử dụng để hiển thị múi giờ tương ứng:

The following information has been given:

```

Brazil
Brazil (southeast: GO, DF, MG, ES, RJ, SP, PR, SC, RS)

```

```

Therefore TZ='America/Sao_Paulo' will be used.
Selected time is now:   sex out 18 18:47:07 -03 2019.
Universal Time is now: sex out 18 21:47:07 UTC 2019.
Is the above information OK?
1) Yes
2) No
#? 1

```

You can make this change permanent for yourself by appending the line

```
TZ='America/Sao_Paulo'; export TZ
```

to the file `/.profile` in your home directory; then log out and log in again.

Here is that TZ value again, this time on standard output so that you can use the `/usr/bin/tzselect` command in shell scripts:

```
America/Sao_Paulo
```

Tên múi giờ kết quả `America/Sao_Paulo` cũng có thể được sử dụng làm nội dung của `/etc/timezone` để thông báo múi giờ mặc định cho hệ thống:

```
$ cat /etc/timezone
```

```
America/Sao_Paulo
```

Như đã nêu trong đầu ra của `tzselect`, biến môi trường `TZ` sẽ xác định múi giờ cho phiên vỏ, bất kể múi giờ mặc định của hệ thống là gì. Việc thêm dòng `TZ='America/Sao_Paulo'; export TZ` vào tệp `~/.profile` sẽ đặt `America/Sao_Paulo` làm múi giờ cho các phiên trong tương lai của người dùng. Biến `TZ` cũng có thể được sửa đổi tạm thời trong phiên hiện tại để hiển thị thời gian ở múi giờ khác:

```
$ env TZ='Africa/Cairo' date
Mon Oct 21 15:45:21 EET 2019
```

Trong ví dụ, lệnh `env` sẽ chạy lệnh đã cho trong một phiên vỏ phụ mới với cùng các biến môi trường của phiên hiện tại ngoại trừ biến `TZ` được sửa đổi bởi đối số `TZ='Africa/Cairo'`.

Quy ước Giờ mùa Hè

Nhiều khu vực sẽ áp dụng quy ước giờ mùa hè cho một khoảng thời gian trong năm — khi đồng hồ thường được điều chỉnh chênh một giờ — điều đó có thể khiến hệ thống bị định cấu hình sai và báo cáo sai thời gian trong mùa đó.

Tệp `/etc/localtime` chứa dữ liệu được hệ điều hành sử dụng để điều chỉnh đồng hồ cho phù hợp. Các hệ thống Linux tiêu chuẩn có các tệp cho tất cả các múi giờ trong thư mục `/usr/share/zoneinfo/`, vì vậy nên `/etc/localtime` chỉ là một liên kết tượng trưng đến tệp dữ liệu thực tế bên trong thư mục đó. Các tệp trong `/usr/share/zoneinfo/` được sắp xếp theo tên của múi giờ tương ứng nên tệp dữ liệu cho múi giờ `America/Sao_Paulo` sẽ là `/usr/share/zoneinfo/America/Sao_Paulo`

Vì các định nghĩa về quy ước giờ mùa hè có thể thay đổi nên quan trọng nhất là người dùng phải luôn cập nhật các tệp ở `/usr/share/zoneinfo/`. Lệnh nâng cấp của công cụ quản lý gói do nhà phân phối cung cấp sẽ cập nhật chúng mỗi khi có phiên bản mới.

Mã hóa Ngôn ngữ và Ký tự

Hệ thống Linux có thể hoạt động với nhiều ngôn ngữ và các mã hóa ký tự không phải của phương Tây - các định nghĩa được gọi là *ngôn ngữ địa phương*. Cấu hình ngôn ngữ địa phương cơ bản nhất chính là định nghĩa của biến môi trường `LANG` mà từ đó, hầu hết các chương trình vỏ sẽ xác định ngôn ngữ để sử dụng.

Nội dung của biến `LANG` sẽ tuân theo định dạng `ab_CD`, trong đó, `ab` là mã ngôn ngữ và `CD` là mã vùng. Mã ngôn ngữ phải tuân theo tiêu chuẩn ISO-639 và mã vùng phải tuân theo tiêu chuẩn ISO-

3166. Ví dụ: một hệ thống được định cấu hình để sử dụng tiếng Bồ Đào Nha Brazil phải có biến LANG được xác định thành `pt_BR.UTF-8`:

```
$ echo $LANG
pt_BR.UTF-8
```

Như đã thấy trong kết quả đầu ra của ví dụ, biến LANG cũng có chứa mã hóa ký tự dành cho hệ thống. ASCII (viết tắt của *Chuẩn mã trao đổi thông tin Hoa Kỳ*) là tiêu chuẩn mã hóa ký tự được sử dụng rộng rãi đầu tiên cho giao tiếp điện tử. Tuy nhiên, vì ASCII có phạm vi giá trị số khả dụng rất hạn chế và được dựa trên bảng chữ cái tiếng Anh nên nó không chứa các ký tự được sử dụng bởi các ngôn ngữ khác hoặc các bộ ký hiệu không phải chữ cái tiếng Anh mở rộng. Mã hóa UTF-8 là *Tiêu chuẩn Unicode* dành cho các ký tự phương Tây thông thường cùng với nhiều ký hiệu không thông thường khác. Như đã được đơn vị duy trì *Tiêu chuẩn Unicode* là *Hiệp hội Unicode* khẳng định, nó phải được áp dụng theo mặc định để đảm bảo khả năng tương thích giữa các nền tảng máy tính:

Tiêu chuẩn Unicode cung cấp một số nhận dạng cho mỗi ký tự bất kể trong nền tảng, thiết bị, ứng dụng hay ngôn ngữ nào. Nó đã được tất cả các nhà cung cấp phần mềm hiện đại áp dụng và hiện cho phép dữ liệu được truyền qua nhiều nền tảng, thiết bị và ứng dụng khác nhau mà không gặp lỗi. Hỗ trợ Unicode tạo nền tảng cho việc thể hiện ngôn ngữ và ký hiệu trong tất cả các hệ điều hành, công cụ tìm kiếm, trình duyệt, máy tính xách tay và điện thoại thông minh chính — cộng với Internet và World Wide Web (URL, HTML, XML, CSS, JSON, v.v. .). (...) Tiêu chuẩn Unicode và sự sẵn có của các công cụ hỗ trợ nó là một trong những xu hướng công nghệ phần mềm toàn cầu quan trọng nhất hiện nay.

— The Unicode Consortium, What is Unicode?

Một số hệ thống vẫn có thể sử dụng các tiêu chuẩn do ISO xác định — như tiêu chuẩn ISO-8859-1 — để mã hóa các ký tự không phải ASCII. Tuy nhiên, các tiêu chuẩn mã hóa ký tự như vậy không nên được sử dụng nữa và nên được thay thế theo các tiêu chuẩn mã hóa Unicode. Nhìn chung, tất cả các hệ điều hành chính đều có xu hướng áp dụng tiêu chuẩn Unicode theo mặc định.

Cài đặt ngôn ngữ trên toàn hệ thống được định cấu hình trong tệp `/etc/locale.conf`. Biến LANG và các biến liên quan đến ngôn ngữ địa phương khác được gán trong tệp này giống như một biến vỏ thông thường. Ví dụ:

```
$ cat /etc/locale.conf
LANG=pt_BR.UTF-8
```

Người dùng có thể sử dụng cấu hình ngôn ngữ tùy chỉnh bằng cách xác định lại biến môi trường

LANG. Nó có thể được thực hiện chỉ cho phiên hiện tại hoặc cho các phiên trong tương lai bằng cách thêm định nghĩa mới vào hồ sơ Bash của người dùng trong `~/.bash_profile` hoặc `~/.profile`. Tuy nhiên, cho đến khi người dùng đăng nhập, ngôn ngữ hệ thống mặc định vẫn sẽ được các chương trình độc lập với người dùng sử dụng (như màn hình đăng nhập của trình quản lý hiển thị).

TIP Lệnh `localectl` có sẵn trên các hệ thống sử dụng `systemd` làm trình quản lý hệ thống và cũng có thể được sử dụng để truy vấn và thay đổi ngôn ngữ hệ thống. Ví dụ: `localectl set-locale LANG=en_US.UTF-8`.

Ngoài biến LANG, các biến môi trường khác cũng có ảnh hưởng đến các khía cạnh ngôn ngữ cụ thể (ví dụ như việc sử dụng ký hiệu tiền tệ nào hoặc dấu phân cách hàng nghìn chính xác cho các số):

LC_COLLATE

Đặt thứ tự bảng chữ cái. Một trong những mục đích của nó là xác định thứ tự các tệp và thư mục được liệt kê.

LC_CTYPE

Đặt phương thức xử lý các bộ ký tự nhất định. Ví dụ, nó sẽ xác định những ký tự nào được coi là *chữ hoa* hoặc *chữ thường*.

LC_MESSAGES

Đặt ngôn ngữ để hiển thị thông báo chương trình (hầu hết là các chương trình GNU).

LC_MONETARY

Đặt đơn vị tiền tệ và định dạng tiền tệ.

LC_NUMERIC

Đặt định dạng số cho các giá trị phi tiền tệ. Mục đích chính của nó là xác định dấu phân cách phần nghìn và thập phân.

LC_TIME

Đặt định dạng ngày và giờ.

LC_PAPER

Đặt khổ giấy tiêu chuẩn.

LC_ALL

Ghi đè tất cả các biến khác, bao gồm cả LANG.

Lệnh `locale` sẽ hiển thị tất cả các biến được xác định trong cấu hình ngôn ngữ hiện tại:

\$ locale

```
LANG=pt_BR.UTF-8
LC_CTYPE="pt_BR.UTF-8"
LC_NUMERIC=pt_BR.UTF-8
LC_TIME=pt_BR.UTF-8
LC_COLLATE="pt_BR.UTF-8"
LC_MONETARY=pt_BR.UTF-8
LC_MESSAGES="pt_BR.UTF-8"
LC_PAPER=pt_BR.UTF-8
LC_NAME=pt_BR.UTF-8
LC_ADDRESS=pt_BR.UTF-8
LC_TELEPHONE=pt_BR.UTF-8
LC_MEASUREMENT=pt_BR.UTF-8
LC_IDENTIFICATION=pt_BR.UTF-8
LC_ALL=
```

Biến không xác định duy nhất là `LC_ALL` và có thể được sử dụng để tạm thời ghi đè tất cả các cài đặt ngôn ngữ khác. Ví dụ sau đây cho thấy lệnh `date` — chạy trong hệ thống được định cấu hình thành ngôn ngữ `pt_BR.UTF-8` — sẽ sửa đổi đầu ra của nó để tuân thủ biến `LC_ALL` mới:

\$ date

```
seg out 21 10:45:21 -03 2019
$ env LC_ALL=en_US.UTF-8 date
Mon Oct 21 10:45:21 -03 2019
```

Việc sửa đổi biến `LC_ALL` sẽ làm cho cả hai chữ viết tắt cho tên ngày trong tuần và tên tháng được hiển thị bằng tiếng Anh Mỹ (`en_US`). Tuy nhiên, chúng ta không bắt buộc phải đặt cùng một ngôn ngữ cho tất cả các biến. Ví dụ: có thể xác định ngôn ngữ thành `pt_BR` và định dạng số (`LC_NUMERIC`) đặt theo tiêu chuẩn Mỹ.

Một số cài đặt địa phương hóa sẽ thay đổi cách chương trình xử lý các định dạng số và thứ tự bảng chữ cái. Ví dụ: trong khi các chương trình thông thường thường sẽ chọn chính xác một ngôn ngữ chung cho các tình huống như vậy thì các tệp lệnh có thể sẽ không hoạt động được như mong muốn khi cố gắng sắp xếp một danh sách các mục chính xác theo thứ tự bảng chữ cái. Vì lý do này, người dùng nên đặt biến môi trường `LANG` thành ngôn ngữ C chung (như trong `LANG=C`) để tệp lệnh tạo ra các kết quả rõ ràng, bất kể các định nghĩa địa phương hóa được sử dụng trong hệ thống nơi nó được thực thi là gì. Ngôn ngữ địa phương C sẽ chỉ tiến hành một so sánh về byte đơn giản nên nó cũng sẽ hoạt động tốt hơn các ngôn ngữ khác.

Chuyển đổi Mã hóa

Văn bản có thể xuất hiện với các ký tự khó diễn giải khi hiển thị trên hệ thống có cấu hình mã hóa ký tự khác với hệ thống nơi văn bản được tạo. Lệnh `iconv` có thể được sử dụng để giải quyết vấn đề này bằng cách chuyển đổi tệp từ mã hóa ký tự gốc sang mã mong muốn. Ví dụ: để chuyển đổi một tệp có tên `origin.txt` từ mã hóa ISO-8859-1 sang tệp có tên `converted.txt` sử dụng mã hóa UTF-8, chúng ta có thể sử dụng lệnh sau:

```
$ iconv -f ISO-8859-1 -t UTF-8 original.txt > converted.txt
```

Tùy chọn `-f ISO-8859-1` (hoặc `--from-code=ISO-8859-1`) sẽ đặt mã hóa của tệp gốc và tùy chọn `-t UTF-8` (hoặc `--to-code=UTF-8`) sẽ đặt mã hóa cho tệp được chuyển đổi. Tất cả mã hóa được hỗ trợ bởi lệnh `iconv` đều có thể được liệt kê bằng lệnh `iconv -l` hoặc `iconv --list`. Thay vì sử dụng chuyển hướng đầu ra như trong ví dụ, tùy chọn `-o converted.txt` hoặc `--output converted.txt` cũng có thể được sử dụng.

Bài tập Hướng dẫn

1. Dựa trên kết quả đầu ra sau của lệnh `date`, múi giờ của hệ thống theo ký hiệu GMT là gì?

```
$ date
Mon Oct 21 18:45:21 +05 2019
```

2. Liên kết tương trưng `/etc/localtime` nên trỏ tới tệp nào để biến `Europe/Brussels` thành giờ cục bộ mặc định của hệ thống?

3. Các ký tự trong tệp văn bản có thể sẽ không được hiển thị chính xác trong hệ thống có mã hóa ký tự khác với mã hóa được sử dụng trong tài liệu văn bản. Làm cách nào để có thể sử dụng lệnh `iconv` để chuyển đổi tệp được mã hóa `WINDOWS-1252 old.txt` thành tệp `new.txt` sử dụng mã hóa `UTF-8`?

Bài tập Mở rộng

1. Lệnh nào sẽ biến Pacific/Auckland thành múi giờ mặc định cho phiên vỏ hiện tại?

2. Lệnh `uptime` sẽ hiển thị *tải lượng trung bình* của hệ thống ở dạng phân số bên cạnh nhiều thông tin khác. Nó sử dụng cài đặt ngôn ngữ hiện tại để quyết định xem dấu phân cách vị trí thập phân là dấu chấm hay dấu phẩy. Ví dụ: nếu ngôn ngữ hiện tại được đặt thành `de_DE.UTF-8` (ngôn ngữ tiêu chuẩn của Đức), `uptime` sẽ sử dụng dấu phẩy làm dấu phân cách. Biết rằng trong tiếng Anh Mỹ, dấu chấm được dùng làm dấu phân cách, lệnh nào sẽ khiến `uptime` hiển thị các phân số bằng dấu chấm thay vì dấu phẩy trong phần còn lại của phiên hiện tại?

3. Lệnh `iconv` sẽ thay thế tất cả các ký tự bên ngoài bộ ký tự đích bằng dấu chấm hỏi. Nếu `//TRANSLIT` được thêm vào mã hóa đích, các ký tự không được biểu thị trong bộ ký tự đích sẽ được thay thế (chuyển ngữ) bằng một hoặc nhiều ký tự trông giống như vậy. Làm cách nào để có thể sử dụng phương pháp này để chuyển đổi tệp văn bản UTF-8 có tên `readme.txt` thành tệp ASCII đơn giản có tên `ascii.txt`?

Tóm tắt

Bài học này đã nói về cách thiết lập hệ thống Linux để hoạt động với cài đặt thời gian và ngôn ngữ tùy chỉnh. Các khái niệm và cài đặt mã hóa ký tự cũng được đề cập tới vì chúng rất quan trọng trong việc hiển thị chính xác nội dung văn bản. Bài học đã đi qua các chủ đề sau:

- Cách hệ thống Linux chọn ngôn ngữ để hiển thị thông báo vờ.
- Hiểu về sự ảnh hưởng của múi giờ đến giờ cục bộ như thế nào.
- Cách xác định múi giờ thích hợp và sửa đổi cài đặt hệ thống cho phù hợp.
- Mã hóa ký tự là gì và cách chuyển đổi giữa chúng.

Các lệnh và quy trình đã được giải quyết là:

- Các biến môi trường liên quan đến ngôn ngữ địa phương và thời gian, chẳng hạn như `LC_ALL`, `LANG` và `TZ`.
- `/etc/timezone`
- `/etc/localtime`
- `/usr/share/zoneinfo/`
- `locale`
- `tzselect`
- `timedatectl`
- `date`
- `iconv`

Đáp án Bài tập Hướng dẫn

1. Dựa trên kết quả đầu ra sau của lệnh `date`, múi giờ của hệ thống theo ký hiệu GMT là gì?

```
$ date
Mon Oct 21 18:45:21 +05 2019
```

Câu trả lời là múi giờ `Etc/GMT+5`.

2. Liên kết tượng trưng `/etc/localtime` nên trỏ tới tệp nào để biến `Europe/Brussels` thành giờ cục bộ mặc định của hệ thống?

Liên kết `/etc/localtime` phải trỏ đến `/usr/share/zoneinfo/Europe/Brussels`.

3. Các ký tự trong tệp văn bản có thể sẽ không được hiển thị chính xác trong hệ thống có mã hóa ký tự khác với mã hóa được sử dụng trong tài liệu văn bản. Làm cách nào để có thể sử dụng lệnh `iconv` để chuyển đổi tệp được mã hóa `WINDOWS-1252 old.txt` thành tệp `new.txt` sử dụng mã hóa `UTF-8`?

Lệnh `iconv -f WINDOWS-1252 -to UTF-8 -to new.txt old.txt` sẽ thực hiện chuyển đổi như mong muốn.

Đáp án Bài tập Mở rộng

1. Lệnh nào sẽ biến Pacific/Auckland thành múi giờ mặc định cho phiên vỏ hiện tại?

```
export TZ=Pacific/Auckland
```

2. Lệnh `uptime` sẽ hiển thị *tải lượng trung bình* của hệ thống ở dạng phân số bên cạnh nhiều thông tin khác. Nó sử dụng cài đặt ngôn ngữ hiện tại để quyết định xem dấu phân cách vị trí thập phân là dấu chấm hay dấu phẩy. Ví dụ: nếu ngôn ngữ hiện tại được đặt thành `de_DE.UTF-8` (ngôn ngữ tiêu chuẩn của Đức), `uptime` sẽ sử dụng dấu phẩy làm dấu phân cách. Biết rằng trong tiếng Anh Mỹ, dấu chấm được dùng làm dấu phân cách, lệnh nào sẽ khiến `uptime` hiển thị các phân số bằng dấu chấm thay vì dấu phẩy trong phần còn lại của phiên hiện tại?

```
Lệnh export LC_NUMERIC=en_US.UTF-8 hoặc export LC_ALL=en_US.UTF-8.
```

3. Lệnh `iconv` sẽ thay thế tất cả các ký tự bên ngoài bộ ký tự đích bằng dấu chấm hỏi. Nếu `//TRANSLIT` được thêm vào mã hóa đích, các ký tự không được biểu thị trong bộ ký tự đích sẽ được thay thế (chuyển ngữ) bằng một hoặc nhiều ký tự trông giống như vậy. Làm cách nào để có thể sử dụng phương pháp này để chuyển đổi tệp văn bản UTF-8 có tên `readme.txt` thành tệp ASCII đơn giản có tên `ascii.txt`?

```
Lệnh iconv -f UTF-8 -t ASCII//TRANSLIT -o ascii.txt readme.txt sẽ thực hiện chuyển đổi như mong muốn.
```



Chủ đề 108: Dịch vụ Hệ thống thiết yếu



108.1 Duy trì Thời gian Hệ thống

Tham khảo các mục tiêu LPI

LPIC-1 version 5.0, Exam 102, Objective 108.1

Khối lượng

3

Các lĩnh vực kiến thức chính

- Đặt ngày và giờ hệ thống.
- Đặt đồng hồ phần cứng về thời gian chuẩn trong UTC.
- Cấu hình múi giờ chính xác.
- Cấu hình NTP cơ bản sử dụng ntpd và chrony.
- Biết sử dụng dịch vụ pool.ntp.org.
- Hiểu về lệnh ntpq.

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- `/usr/share/zoneinfo/`
- `/etc/timezone`
- `/etc/localtime`
- `/etc/ntp.conf`
- `/etc/chrony.conf`
- `date`
- `hwclock`
- `timedatectl`

- ntpd
- ntpdate
- chronyc
- pool.ntp.org



108.1 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	108 Dịch vụ Hệ thống thiết yếu
Mục tiêu:	108.1 Duy trì Thời gian Hệ thống
Bài:	1 trên 2

Giới thiệu

Giờ hiện hành là một yếu tố vô cùng quan trọng đối với tin học hiện đại. Tuy nhiên, việc triển khai căn chỉnh thời gian lại phức tạp một cách đáng kinh ngạc. Việc căn chỉnh thời gian có vẻ không quá quan trọng đối với người dùng cuối nhưng các hệ thống cần có khả năng xử lý một cách hiệu quả các tính năng riêng biệt và các trường hợp phức tạp. Múi giờ không mang tính tĩnh nhưng lại có thể bị thay đổi bởi một quyết định hành chính hoặc chính trị (ví dụ như một quốc gia cũng có thể chọn ngừng thực hiện Quy ước giờ mùa hè). Bất kỳ chương trình nào cũng phải có khả năng xử lý những thay đổi đó một cách hợp lý. Rất may mắn cho các quản trị viên hệ thống là các giải pháp căn chỉnh thời gian trên hệ điều hành Linux rất mạnh mẽ, hoàn thiện và thường có thể hoạt động mà không cần có sự can thiệp của con người.

Khi một máy tính Linux khởi động, nó sẽ bắt đầu căn chỉnh thời gian. Chúng ta sẽ gọi nó là *đồng hồ hệ thống* vì nó sẽ được hệ điều hành cập nhật. Ngoài ra, các máy tính hiện đại cũng sẽ có một *đồng hồ phần cứng* hoặc một *đồng hồ thời gian thực*. Đồng hồ phần cứng này thường là một tính năng của bo mạch chủ và sẽ căn chỉnh thời gian bất kể máy tính có đang chạy hay không. Trong quá trình khởi động, thời gian hệ thống sẽ được đặt từ đồng hồ phần cứng, nhưng đa phần hai đồng hồ này sẽ chạy độc lập với nhau. Trong bài học này, chúng ta sẽ thảo luận về các phương thức tương tác với cả đồng hồ hệ thống và đồng hồ phần cứng.

Trên hầu hết các hệ thống Linux hiện đại, thời gian hệ thống và thời gian phần cứng đều sẽ được đồng bộ hóa với *thời gian mạng* được triển khai bởi *Giao thức thời gian mạng* (NTP). Trong phần lớn các trường hợp, cấu hình duy nhất mà người dùng phổ thông sẽ phải thực hiện là đặt múi giờ của họ và NTP sẽ lo phần còn lại. Tuy nhiên, chúng ta sẽ tìm hiểu về một số phương thức làm việc với thời gian theo cách thủ công tại đây và các chi tiết cụ thể về cách định cấu hình thời gian mạng sẽ được đề cập tới trong bài học tiếp theo.

Giờ Địa phương và Giờ Quốc tế

Đồng hồ hệ thống sẽ được đặt thành Giờ phối hợp quốc tế (UTC) - tức giờ địa phương tại Greenwich, Vương quốc Anh. Thông thường thì người dùng nào cũng sẽ muốn biết *giờ địa phương* của họ. Giờ địa phương sẽ được tính bằng cách áp dụng một *phần bù* dựa trên múi giờ cùng với Quy ước giờ mùa hè lên UTC. Chúng ta có thể tránh được rất nhiều vấn đề phức tạp bằng cách sử dụng phương thức này.

Đồng hồ hệ thống có thể được đặt thành giờ UTC hoặc giờ địa phương, nhưng người dùng thường được khuyến khích đặt thành giờ UTC.

Ngày

`date` (ngày) là một tiện ích cốt lõi được sử dụng chỉ đơn giản là để in giờ địa phương:

```
$ date
Sun Nov 17 12:55:06 EST 2019
```

Việc sửa đổi các tùy chọn của lệnh `date` sẽ thay đổi định dạng của đầu ra.

Ví dụ: người dùng có thể sử dụng `date -u` để xem thời gian UTC hiện tại.

```
$ date -u
Sun Nov 17 18:02:51 UTC 2019
```

Một số tùy chọn thường được sử dụng khác sẽ trả về giờ địa phương tuân thủ theo một định dạng RFC được chấp nhận:

-I

Ngày/giờ ở định dạng ISO 8601. Việc thêm `date (-Idate)` sẽ chỉ giới hạn đầu ra cho ngày. Các định dạng khác là `hours` (giờ), `minutes` (phút), `seconds` (giây) và `ns` cho nanoseconds (nano giây).

-R

Trả về ngày và giờ ở định dạng RFC 5322.

--rfc-3339

Trả về ngày và giờ ở định dạng RFC 3339.

Người dùng có thể tùy chỉnh định dạng của `date` bằng các chuỗi được chỉ định trong trang hướng dẫn. Ví dụ: thời gian hiện tại có thể được định dạng theo thời gian Unix:

```
$ date +%s
1574014515
```

Từ trang hướng dẫn của `date`, chúng ta có thể thấy rằng `%s` được dùng để nói đến thời gian Unix.

Thời gian Unix được sử dụng nội bộ trên hầu hết các hệ thống giống Unix. Nó sẽ lưu trữ thời gian UTC dưới dạng số giây kể từ *Kỷ nguyên* (Epoch - được xác định là ngày 1 tháng 1 năm 1970).

NOTE

Số bit cần thiết để lưu trữ thời gian Unix ở thời điểm hiện tại là 32 bit. Điều này sẽ gây ra một vấn đề trong tương lai khi 32 bit không còn đủ để chứa thời gian hiện tại ở định dạng Unix. Điều này sẽ gây ra các sự cố nghiêm trọng cho mọi hệ thống Linux 32 bit. May mắn thay, điều này sẽ không xảy ra cho đến ngày 19 tháng 1 năm 2038.

Bằng cách sử dụng các chuỗi này, chúng ta có thể định dạng ngày và giờ ở hầu hết mọi định dạng mà bất kỳ ứng dụng nào yêu cầu. Tất nhiên, trong hầu hết các trường hợp, tốt hơn hết là người dùng nên tuân theo một tiêu chuẩn được chấp nhận.

Ngoài ra, `date --date` có thể được sử dụng để định dạng một khung thời gian không phải là thời gian hiện tại. Ví dụ như trong trường hợp này, người dùng có thể chỉ định ngày áp dụng cho hệ thống bằng thời gian Unix:

```
$ date --date='@1564013011'
Wed Jul 24 20:03:31 EDT 2019
```

Việc sử dụng tùy chọn `--debug` có thể sẽ rất hữu ích trong việc đảm bảo ngày sẽ được phân tích cú pháp thành công. Hãy quan sát điều gì sẽ xảy ra khi ta truyền một ngày hợp lệ vào lệnh:

```
$ date --debug --date="Fri, 03 Jan 2020 14:00:17 -0500"
date: parsed day part: Fri (day ordinal=0 number=5)
date: parsed date part: (Y-M-D) 2020-01-03
```

```
date: parsed time part: 14:00:17 UTC-05
date: input timezone: parsed date/time string (-05)
date: using specified time as starting value: '14:00:17'
date: warning: day (Fri) ignored when explicit dates are given
date: starting date/time: '(Y-M-D) 2020-01-03 14:00:17 TZ=-05'
date: '(Y-M-D) 2020-01-03 14:00:17 TZ=-05' = 1578078017 epoch-seconds
date: timezone: system default
date: final: 1578078017.000000000 (epoch-seconds)
date: final: (Y-M-D) 2020-01-03 19:00:17 (UTC)
date: final: (Y-M-D) 2020-01-03 14:00:17 (UTC-05)
```

Đây có thể là một công cụ hữu ích trong việc khắc phục sự cố cho một ứng dụng tạo ngày.

Đồng hồ Phần cứng

Người dùng có thể chạy lệnh `hwclock` để xem thời gian được duy trì trên đồng hồ thời gian thực. Lệnh này sẽ yêu cầu đặc quyền nâng cao, vì vậy nên chúng ta sẽ sử dụng `sudo` để gọi lệnh trong trường hợp này:

```
$ sudo hwclock
2019-11-20 11:31:29.217627-05:00
```

Tùy chọn `--verbose` sẽ trả về đầu ra kết quả chi tiết hơn và có thể sẽ hữu ích trong việc khắc phục sự cố:

```
$ sudo hwclock --verbose
hwclock from util-linux 2.34
System Time: 1578079387.976029
Trying to open: /dev/rtc0
Using the rtc interface to the clock.
Assuming hardware clock is kept in UTC time.
Waiting for clock tick...
...got clock tick
Time read from Hardware Clock: 2020/01/03 19:23:08
Hw clock time : 2020/01/03 19:23:08 = 1578079388 seconds since 1969
Time since last adjustment is 1578079388 seconds
Calculated Hardware Clock drift is 0.000000 seconds
2020-01-03 14:23:07.948436-05:00
```

Hãy lưu ý về `Calculated Hardware Clock drift` (Phần trôi được tính của Đồng hồ Phần cứng). Đầu ra này có thể cho chúng ta biết liệu thời gian hệ thống và thời gian phần cứng có lệch nhau

hay không.

timedatectl

`timedatectl` là một lệnh có thể được sử dụng để kiểm tra trạng thái chung về ngày và giờ bao gồm cả thời gian mạng đã được đồng bộ hóa hay chưa (Giao thức thời gian mạng sẽ được đề cập tới trong bài học tiếp theo).

Theo mặc định, `timedatectl` sẽ trả về thông tin tương tự như `date` nhưng có thêm thời gian RTC (phần cứng) cũng như trạng thái của dịch vụ NTP:

```
$ timedatectl
      Local time: Thu 2019-12-05 11:08:05 EST
      Universal time: Thu 2019-12-05 16:08:05 UTC
      RTC time: Thu 2019-12-05 16:08:05
      Time zone: America/Toronto (EST, -0500)
System clock synchronized: yes
      NTP service: active
      RTC in local TZ: no
```

Cài đặt thời gian bằng cách sử dụng timedatectl

Nếu NTP không có sẵn, chúng ta nên sử dụng `timedatectl` thay vì `date` hoặc `hwclock` để đặt thời gian:

```
# timedatectl set-time '2011-11-25 14:00:00'
```

Quy trình này cũng tương tự như của `date`. Người dùng cũng có thể đặt thời gian độc lập với ngày bằng định dạng HH:MM:SS.

Đặt múi giờ bằng timedatectl

`timedatectl` là cách ưa thích để đặt múi giờ địa phương trên các hệ thống Linux dựa trên `systemd` khi không có GUI tồn tại. `timedatectl` sẽ liệt kê các múi giờ có thể có và sau đó chúng ta có thể đặt bằng cách sử dụng một trong các múi giờ đó làm đối số.

Đầu tiên, chúng ta sẽ liệt kê các múi giờ có thể có:

```
$ timedatectl list-timezones
Africa/Abidjan
```

```
Africa/Accra
Africa/Algiers
Africa/Bissau
Africa/Cairo
...
```

Danh sách các múi giờ có thể có sẽ rất dài; do đó, chúng ta nên sử dụng lệnh `grep` trong trường hợp này.

Tiếp theo, chúng ta có thể đặt múi giờ bằng cách sử dụng một trong các kết quả được trả về:

```
$ timedatectl set-timezone Africa/Cairo
$ timedatectl
           Local time: Thu 2019-12-05 18:18:10 EET
           Universal time: Thu 2019-12-05 16:18:10 UTC
           RTC time: Thu 2019-12-05 16:18:10
           Time zone: Africa/Cairo (EET, +0200)
System clock synchronized: yes
           NTP service: active
           RTC in local TZ: no
```

Hãy nhớ rằng tên của múi giờ phải chính xác. Ví dụ: `Africa/Cairo` sẽ thay đổi được múi giờ, nhưng `cairo` hoặc `africa/cairo` thì không.

Vô hiệu hóa NTP bằng cách sử dụng `timedatectl`

Trong một số trường hợp, người dùng có thể sẽ phải tắt NTP. Việc này có thể được thực hiện bằng cách sử dụng `systemctl` nhưng chúng ta sẽ minh họa bằng `timedatectl`:

```
# timedatectl set-ntp no
$ timedatectl
           Local time: Thu 2019-12-05 18:19:04 EET Universal time: Thu 2019-12-05 16:19:04
UTC
           RTC time: Thu 2019-12-05 16:19:04
           Time zone: Africa/Cairo (EET, +0200)
           NTP enabled: no
           NTP synchronized: no
           RTC in local TZ: no
           DST active: n/a
```

Đặt múi giờ không sử dụng `timedatectl`

Đặt thông tin múi giờ là bước tiêu chuẩn trong cài đặt Linux trên máy mới. Nếu có quá trình cài đặt đồ họa, rất có thể việc này sẽ được tự động xử lý mà không cần người dùng can thiệp.

Thư mục `/usr/share/zoneinfo` sẽ chứa các thông tin có thể có về các múi giờ khác nhau. Trong thư mục `zoneinfo` có các thư mục con có chứa tên các châu lục cũng như các liên kết tượng trưng khác. Chúng ta nên tìm `zoneinfo` của khu vực bắt đầu từ lục địa.

Các tệp `zoneinfo` có chứa các quy tắc bắt buộc để tính toán chênh lệch thời gian địa phương liên quan đến UTC và chúng cũng rất quan trọng nếu khu vực của người dùng tuân theo Quy ước giờ mùa hè. Nội dung của `/etc/localtime` sẽ được đọc khi Linux cần xác định múi giờ địa phương. Để đặt múi giờ mà không cần sử dụng GUI, người dùng nên tạo một liên kết tượng trưng cho vị trí của họ từ `/usr/share/zoneinfo` đến `/etc/localtime`. Ví dụ:

```
$ ln -s /usr/share/zoneinfo/Canada/Eastern /etc/localtime
```

Sau khi đặt múi giờ chính xác, người dùng được khuyến nghị chạy:

```
# hwclock --systohc
```

Lệnh này sẽ đặt *đồng hồ phần cứng* từ *đồng hồ hệ thống* (nghĩa là đồng hồ thời gian thực sẽ được đặt cùng thời gian với `date`). Hãy lưu ý rằng lệnh này phải được chạy với quyền gốc, trong trường hợp này người dùng đang đăng nhập bằng quyền gốc.

`/etc/timezone` cũng tương tự như `/etc/localtime`. Nó là biểu diễn bằng dữ liệu của múi giờ địa phương và do đó có thể được đọc bằng cách sử dụng `cat`:

```
$ cat /etc/timezone
America/Toronto
```

Hãy lưu ý rằng tệp này không được sử dụng bởi tất cả các bản phân phối Linux.

Cài đặt ngày giờ không sử dụng `timedatectl`

NOTE

Hầu hết các hệ thống Linux hiện đại đều sử dụng `systemd` cho cấu hình và dịch vụ của nó và do đó người dùng không nên sử dụng `date` hoặc `hwclock` để cài đặt thời gian. `systemd` sẽ sử dụng `timedatectl` cho việc này. Tuy nhiên, quan trọng nhất là người dùng phải biết về các lệnh kế thừa này trong trường hợp phải quản trị một

hệ thống cũ.

Sử dụng ngày

`date` có tùy chọn để đặt thời gian hệ thống là `--set` hoặc `-s` để đặt ngày và giờ. Chúng ta cũng có thể chọn sử dụng `--debug` để xác minh việc phân tích cú pháp chính xác của lệnh:

```
# date --set="11 Nov 2011 11:11:11"
```

Hãy lưu ý rằng người dùng cần có quyền gốc để đặt ngày ở đây. Chúng ta cũng có thể chọn thay đổi thời gian hoặc ngày một cách độc lập:

```
# date +%Y%m%d -s "20111125"
```

Ở đây chúng ta phải chỉ định các trình tự để chuỗi được phân tích cú pháp một cách chính xác. Ví dụ: `%Y` đề cập đến năm và do đó bốn chữ số đầu tiên `2011` sẽ được hiểu là năm 2011. Tương tự, `%T` là chuỗi thời gian và nó được thể hiện ở đây bằng cách đặt thời gian:

```
# date +%T -s "13:11:00"
```

Sau khi thay đổi thời gian hệ thống, chúng ta cũng nên đặt đồng hồ phần cứng để cả đồng hồ hệ thống và phần cứng đều được đồng bộ hóa:

```
# hwclock --systohc
```

`systohc` (system clock to hardware clock) có nghĩa là “đồng hồ hệ thống đến đồng hồ phần cứng”

Sử dụng hwclock

Thay vì đặt đồng hồ hệ thống và cập nhật đồng hồ phần cứng, chúng ta có thể đảo ngược tiến trình và bắt đầu bằng cách đặt đồng hồ phần cứng:

```
# hwclock --set --date "4/12/2019 11:15:19"
# hwclock
Fri 12 Apr 2019 6:15:19 AM EST -0.562862 seconds
```

Hãy lưu ý rằng theo mặc định, `hwclock` sẽ giả định thời gian UTC nhưng vẫn trả về giờ địa

phương theo mặc định.

Sau khi cài đặt đồng hồ phần cứng, chúng ta sẽ cần cập nhật đồng hồ hệ thống từ nó. `hwclock --hctosys` (hardware clock to system clock) có thể hiểu là “đồng hồ phần cứng đến đồng hồ hệ thống”.

```
# hwclock --hctosys
```

Bài tập Hướng dẫn

1. Hãy cho biết các lệnh sau đây đang hiển thị hoặc sửa đổi *thời gian hệ thống* hay *thời gian phần cứng*:

Lệnh	Hệ thống	Phần cứng	Cả hai
<code>date -u</code>			
<code>hwclock --set --date "12:00:00"</code>			
<code>timedatectl</code>			
<code>timedatectl grep RTC</code>			
<code>hwclock --hctosys</code>			
<code>date +%T -s "08:00:00"</code>			
<code>timedatectl set- time 1980-01-10</code>			

2. Hãy quan sát kết quả đầu ra sau và sửa định dạng của đối số để lệnh thành công:

```
$ date --debug --date "20/20/12 0:10 -3"

date: warning: value 20 has less than 4 digits. Assuming MM/DD/YY[YY]
date: parsed date part: (Y-M-D) 0002-20-20
date: parsed time part: 00:10:00 UTC-03
date: input timezone: parsed date/time string (-03)
date: using specified time as starting value: '00:10:00'
date: error: invalid date/time value:
date:   user provided time: '(Y-M-D) 0002-20-20 00:10:00 TZ=-03'
date:   normalized time: '(Y-M-D) 0003-08-20 00:10:00 TZ=-03'
date:   -----
date:   possible reasons:
date:     numeric values overflow;
date:     incorrect timezone
date: invalid date '20/20/2 0:10 -3'
```

3. Bằng cách sử dụng lệnh `date` và các chuỗi, hãy viết lệnh để tháng của hệ thống được đặt thành

Tháng Hai và giữ nguyên phần còn lại của ngày và giờ.

4. Giả sử lệnh trên thành công, hãy sử dụng `hwclock` để đặt đồng hồ phần cứng từ đồng hồ hệ thống.

5. Có một địa điểm được gọi là `eucla`. Nó là một phần của lục địa nào? Hãy sử dụng lệnh `grep` để tìm hiểu.

6. Hãy đặt múi giờ hiện tại của bạn thành múi giờ `eucla`.

Bài tập Mở rộng

1. Phương pháp cài đặt thời gian nào là tối ưu? Trong trường hợp nào phương pháp thức này có thể sẽ không thể thực hiện được?

2. Tại sao có rất nhiều phương pháp để thực hiện cùng một việc (cài đặt thời gian hệ thống)?

3. Sau ngày 19 tháng 1 năm 2038, Giờ hệ thống Linux sẽ yêu cầu một số 64 bit để lưu trữ. Tuy nhiên, có thể chúng ta chỉ cần chọn đặt một “Kỷ nguyên mới”. Ví dụ: lúc nửa đêm ngày 1 tháng 1 năm 2038 có thể được đặt thành Thời gian kỷ nguyên mới là 0. Tại sao điều này lại không trở thành một giải pháp tối ưu?

Tóm tắt

Trong bài học này, chúng ta đã học về:

- Cách hiển thị thời gian ở các định dạng khác nhau từ dòng lệnh.
- Sự khác biệt giữa đồng hồ hệ thống và đồng hồ phần cứng trong Linux.
- Cách đặt đồng hồ hệ thống theo cách thủ công.
- Cách đặt đồng hồ phần cứng theo cách thủ công.
- Cách thay đổi múi giờ của hệ thống.

Các lệnh được sử dụng trong bài học này:

date

Hiển thị hoặc thay đổi đồng hồ hệ thống. Các tùy chọn:

-u

Hiển thị thời gian UTC.

+%s

Sử dụng một chuỗi để hiển thị thời gian Epoch.

--date=

Chỉ định thời gian cụ thể để hiển thị chứ không phải thời gian hiện tại.

--debug

Hiển thị thông báo gỡ lỗi khi phân tích ngày do người dùng nhập.

-s

Đặt đồng hồ hệ thống theo cách thủ công.

hwclock

Hiển thị hoặc thay đổi đồng hồ phần cứng.

--systohc

Sử dụng đồng hồ hệ thống để đặt đồng hồ phần cứng.

--hctosys

Sử dụng đồng hồ phần cứng để đặt đồng hồ hệ thống.

--set --date

Đặt đồng hồ phần cứng theo cách thủ công.

timedatectl

Hiển thị đồng hồ hệ thống và đồng hồ phần cứng cũng như cấu hình NTP trên hệ thống Linux dựa trên systemd.

set-time

Đặt thời gian theo cách thủ công.

list-timezones

Liệt kê các múi giờ có thể có.

set-timezone

Đặt múi giờ theo cách thủ công.

set-ntp

Kích hoạt/vô hiệu hoá NTP.

Đáp án Bài tập Hướng dẫn

1. Hãy cho biết các lệnh sau đây đang hiển thị hoặc sửa đổi *thời gian hệ thống* hay *thời gian phần cứng*:

Lệnh	Hệ thống	Phần cứng	Cả hai
<code>date -u</code>	X		
<code>hwclock --set --date "12:00:00"</code>		X	
<code>timedatectl</code>			X
<code>timedatectl grep RTC</code>		X	
<code>hwclock --hctosys</code>	X		
<code>date +%T -s "08:00:00"</code>	X		
<code>timedatectl set- time 1980-01-10</code>			X

2. Hãy quan sát kết quả đầu ra sau và sửa định dạng của đối số để lệnh thành công:

```
$ date --debug --date "20/20/12 0:10 -3"
```

```
date: warning: value 20 has less than 4 digits. Assuming MM/DD/YY[YY]
date: parsed date part: (Y-M-D) 0002-20-20
date: parsed time part: 00:10:00 UTC-03
date: input timezone: parsed date/time string (-03)
date: using specified time as starting value: '00:10:00'
date: error: invalid date/time value:
date:     user provided time: '(Y-M-D) 0002-20-20 00:10:00 TZ=-03'
date:     normalized time: '(Y-M-D) 0003-08-20 00:10:00 TZ=-03'
date:
date:     possible reasons:
date:     numeric values overflow;
date:     incorrect timezone
date: invalid date '20/20/2 0:10 -3'
```

```
date --debug --set "12/20/20 0:10 -3"
```

3. Bằng cách sử dụng lệnh `date` và các chuỗi, hãy viết lệnh để tháng của hệ thống được đặt thành Tháng Hai và giữ nguyên phần còn lại của ngày và giờ.

```
date +%m -s "2"
```

4. Giả sử lệnh trên thành công, hãy sử dụng `hwclock` để đặt đồng hồ phần cứng từ đồng hồ hệ thống.

```
hwclock -systohc
```

5. Có một địa điểm được gọi là `eucla`. Nó là một phần của lục địa nào? Hãy sử dụng lệnh `grep` để tìm hiểu.

```
timedatectl list-timezones \| grep -i eucla
```

HOẶC

```
grep -ri eucla /usr/share/zoneinfo
```

6. Hãy đặt múi giờ hiện tại của bạn thành múi giờ `eucla`.

```
timedatectl set-timezone 'Australia/Eucla'
```

hoặc

```
ln -s /usr/share/zoneinfo/Australia/Eucla /etc/localtime
```

Đáp án Bài tập Mở rộng

1. Phương pháp cài đặt thời gian nào là tối ưu? Trong trường hợp nào phương pháp thức này có thể sẽ không thể thực hiện được?

Trong hầu hết các bản phân phối Linux, NTP sẽ được kích hoạt theo mặc định và nên được dùng để đặt thời gian hệ thống mà không có sự can thiệp của người dùng. Tuy nhiên, nếu có hệ thống Linux không được kết nối với internet, NTP sẽ không thể truy cập được (ví dụ như hệ thống nhúng Linux chạy trên thiết bị công nghiệp có thể sẽ không có kết nối mạng).

2. Tại sao có rất nhiều phương pháp để thực hiện cùng một việc (cài đặt thời gian hệ thống)?

Vì cài đặt thời gian là yêu cầu của tất cả các hệ thống *nix trong nhiều thập kỷ nên có nhiều phương pháp cũ để cài đặt thời gian vẫn được duy trì.

3. Sau ngày 19 tháng 1 năm 2038, Giờ hệ thống Linux sẽ yêu cầu số 64 bit để lưu trữ. Tuy nhiên, có thể chúng ta chỉ cần chọn đặt một “Kỷ nguyên mới”. Ví dụ: lúc nửa đêm ngày 1 tháng 1 năm 2038 có thể được đặt thành Thời gian kỷ nguyên mới là 0. Tại sao điều này lại không trở thành một giải pháp tối ưu?

Đến năm 2038, phần lớn máy tính sẽ chạy CPU 64 bit và việc sử dụng số 64 bit sẽ không làm giảm hiệu suất theo bất kỳ một cách đáng kể nào. Tuy nhiên, chúng ta không thể ước tính rủi ro của việc “đặt lại” thời gian Kỷ nguyên theo cách như vậy. Có rất nhiều phần mềm cũ có thể bị ảnh hưởng. Ví dụ: các ngân hàng và doanh nghiệp lớn thường có một lượng lớn các chương trình cũ mà họ sử dụng nội bộ. Vì vậy, cũng giống như nhiều kịch bản khác, kịch bản này là một thí nghiệm về sự đánh đổi. Bất kỳ hệ thống 32 bit nào vẫn chạy vào năm 2038 cũng sẽ bị ảnh hưởng do tràn Thời gian Kỷ nguyên, nhưng các phần mềm cũ cũng sẽ bị ảnh hưởng khi giá trị của Epoch bị thay đổi.



108.1 Bài 2

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	108 Dịch vụ Hệ thống thiết yếu
Mục tiêu:	108.1 Duy trì Thời gian Hệ thống
Bài:	2 trên 2

Giới thiệu

Trong khi máy tính cá nhân có thể tự căn chỉnh thời gian chính xác một cách hợp lý thì môi trường mạng và tin học sản xuất yêu cầu thời gian phải được chạy ở mức độ chính xác tuyệt đối. Thời gian chính xác nhất được đo bằng *đồng hồ tham chiếu*, thường là đồng hồ nguyên tử. Thế giới hiện đại đã phát minh ra một hệ thống mà trong đó tất cả các hệ thống máy tính được kết nối internet đều có thể được đồng bộ hóa với các đồng hồ tham chiếu này bằng cách sử dụng *Giao thức thời gian mạng* (NTP). Một hệ thống máy tính có NTP sẽ có thể đồng bộ hóa đồng hồ hệ thống của chúng với thời gian do đồng hồ tham chiếu cung cấp. Nếu thời gian hệ thống và thời gian được đo trên các máy chủ này khác nhau thì máy tính sẽ tăng dần hoặc làm chậm thời gian hệ thống nội bộ của nó cho đến khi thời gian hệ thống khớp với thời gian mạng.

NTP sử dụng cấu trúc phân cấp để phân bổ thời gian. Đồng hồ tham chiếu sẽ được kết nối với các máy chủ ở đầu hệ thống phân cấp. Các máy chủ này là máy *Tầng (Stratum) 1* và mạng công cộng thường không thể truy cập được vào đây. Tuy nhiên, các máy *Tầng 1* lại có thể truy cập được bằng các máy *Tầng 2*; *Tầng 2* lại có thể được truy cập bằng các máy *Tầng 3*, v.v. Trong mạng công cộng, người dùng có thể truy cập máy chủ *Tầng 2* cũng như bất kỳ máy nào thấp hơn trong hệ thống phân cấp. Khi thiết lập NTP cho một mạng lớn, cách tốt nhất là có một số lượng nhỏ máy tính kết nối với máy chủ *Tầng 2+*, sau đó yêu cầu các máy đó cung cấp NTP cho tất cả các máy khác. Bằng

cách này, nhu cầu về máy Tầng 2 có thể được giảm thiểu.

Có một số thuật ngữ quan trọng sẽ xuất hiện khi nói về về NTP. Một số thuật ngữ này được áp dụng trong các lệnh mà chúng ta sẽ sử dụng để theo dõi trạng thái của NTP trên máy của mình:

Phần bù (Offset)

Thuật ngữ này nói đến sự khác biệt tuyệt đối giữa thời gian hệ thống và thời gian NTP. Ví dụ: nếu đồng hồ hệ thống đọc thời gian là 12:00:02 và thời gian NTP là 11:59:58 thì phần bù giữa hai đồng hồ là bốn giây.

Bước nhảy dài (Step)

Nếu khoảng cách thời gian giữa nhà cung cấp NTP và người dùng lớn hơn 128 mili giây thì NTP sẽ thực hiện một thay đổi đáng kể đối với thời gian hệ thống thay vì làm chậm hoặc tăng tốc thời gian hệ thống. Điều này được gọi là *nhảy cóc*.

Xoay (Slew)

Thuật ngữ này nói đến những thay đổi được thực hiện đối với thời gian hệ thống khi phần bù giữa thời gian hệ thống và NTP nhỏ hơn 128 mili giây. Trong trường hợp này, những thay đổi sẽ được thực hiện dần dần. Điều này được gọi là *xoay*.

Thời gian loạn (Insane Time)

Nếu phần bù giữa thời gian hệ thống và thời gian NTP lớn hơn 17 phút thì thời gian hệ thống sẽ được coi là *loạn* và trình nền NTP sẽ không đưa ra bất kỳ một thay đổi nào đối với thời gian hệ thống. Một số thao tác đặc biệt sẽ phải được thực hiện để đưa thời gian của hệ thống về đúng 17 phút so với thời gian chuẩn.

Trôi (Drift)

Thuật ngữ này nói đến hiện tượng hai đồng hồ không đồng bộ theo thời gian. Về cơ bản, nếu hai đồng hồ được đồng bộ hóa ban đầu nhưng sau đó bị lệch theo thời gian thì có nghĩa là hiện tượng trôi đồng hồ đang xảy ra.

Phương sai độ trễ (Jitter)

Thuật ngữ này nói đến mức độ trôi kể từ lần cuối cùng đồng hồ được truy vấn. Vì vậy, nếu lần đồng bộ hóa NTP cuối cùng xảy ra cách đây 17 phút và phần bù giữa nhà cung cấp NTP và người tiêu dùng là 3 mili giây thì 3 mili giây là chính là phương sai độ trễ.

Bây giờ chúng ta sẽ thảo luận về một số cách cụ thể mà Linux sử dụng để triển khai NTP.

timedatectl

Nếu một bản phân phối Linux sử dụng `timedatectl` thì theo mặc định, nó sẽ triển khai một máy

khách *SNTP* thay vì triển khai NTP đầy đủ. Đây là cách triển khai thời gian mạng cho máy ít phức tạp hơn và máy sẽ không phục vụ NTP cho các máy tính được kết nối khác.

Trong trường hợp này, *SNTP* sẽ không hoạt động trừ khi dịch vụ `timesyncd` đang chạy. Như với tất cả các dịch vụ `systemd`, chúng ta có thể xác minh rằng nó đang chạy với:

```
$ systemctl status systemd-timesyncd
● systemd-timesyncd.service - Network Time Synchronization
   Loaded: loaded (/lib/systemd/system/systemd-timesyncd.service; enabled; vendor preset:
   enabled)
   Drop-In: /lib/systemd/system/systemd-timesyncd.service.d
            └─disable-with-time-daemon.conf
   Active: active (running) since Thu 2020-01-09 21:01:50 EST; 2 weeks 1 days ago
     Docs: man:systemd-timesyncd.service(8)
  Main PID: 1032 (systemd-timesyn)
    Status: "Synchronized to time server for the first time 91.189.89.198:123
(ntp.ubuntu.com)."
```

```
   Tasks: 2 (limit: 4915)
  Memory: 3.0M
   CGroup: /system.slice/systemd-timesyncd.service
            └─1032 /lib/systemd/systemd-timesyncd

Jan 11 13:06:18 NeoMex systemd-timesyncd[1032]: Synchronized to time server for the first
time 91.189.91.157:123 (ntp.ubuntu.com).
...
```

Trạng thái đồng bộ hóa *SNTP* của `timedatectl` có thể được xác minh bằng cách sử dụng `show-timesync`:

```
$ timedatectl show-timesync --all
LinkNTPServers=
SystemNTPServers=
FallbackNTPServers=ntp.ubuntu.com
ServerName=ntp.ubuntu.com
ServerAddress=91.189.89.198
RootDistanceMaxUsec=5s
PollIntervalMinUsec=32s
PollIntervalMaxUsec=34min 8s
PollIntervalUsec=34min 8s
NTPMessage={ Leap=0, Version=4, Mode=4, Stratum=2, Precision=-23, RootDelay=8.270ms,
RootDispersion=18.432ms, Reference=91EECB0E, OriginateTimestamp=Sat 2020-01-25 18:35:49 EST,
ReceiveTimestamp=Sat 2020-01-25 18:35:49 EST, TransmitTimestamp=Sat 2020-01-25 18:35:49 EST,
```

```
DestinationTimestamp=Sat 2020-01-25 18:35:49 EST, Ignored=no PacketCount=263, Jitter=2.751ms
}
Frequency=-211336
```

Cấu hình này có thể phù hợp cho hầu hết các trường hợp, nhưng như đã lưu ý trước đó, nó sẽ không đủ nếu người dùng muốn đồng bộ hóa nhiều máy khách trong mạng. Trong trường hợp này, chúng ta nên cài đặt một máy khách NTP đầy đủ.

Trình nền NTP

Thời gian hệ thống được so sánh với thời gian mạng theo lịch trình thông thường. Để tính năng này hoạt động, chúng ta phải có một *trình nền* chạy ngầm. Đối với nhiều hệ thống Linux, tên của trình nền này là `ntpd`. `ntpd` sẽ cho phép một máy không chỉ là *người tiêu thụ thời gian* (nghĩa là có thể đồng bộ hóa đồng hồ của chính nó từ nguồn bên ngoài) mà còn có thể *cung cấp* thời gian cho các máy khác.

Giả sử máy tính của chúng ta được dựa trên `systemd` và nó sử dụng `systemctl` để điều khiển các trình nền. Chúng ta sẽ cài đặt các gói `ntp` bằng trình quản lý gói thích hợp và sau đó đảm bảo rằng trình nền `ntpd` đang chạy bằng cách kiểm tra trạng thái của nó:

```
$ systemctl status ntpd

● ntpd.service - Network Time Service
   Loaded: loaded (/usr/lib/systemd/system/ntpd.service; enabled; vendor preset: disabled)
   Active: active (running) since Fri 2019-12-06 03:27:21 EST; 7h ago
   Process: 856 ExecStart=/usr/sbin/ntpd -u ntp:ntp $OPTIONS (code=exited, status=0/SUCCESS)
   Main PID: 867 (ntpd)
   CGroup: /system.slice/ntpd.service
           └─867 /usr/sbin/ntpd -u ntp:ntp -g
```

Trong một số trường hợp, có thể chúng ta sẽ cần phải khởi động và kích hoạt `ntpd`. Trên hầu hết các máy Linux, việc này được thực hiện bằng:

```
# systemctl enable ntpd && systemctl start ntpd
```

Các truy vấn NTP sẽ xảy ra trên Cổng TCP 123. Nếu NTP không hoạt động, hãy đảm bảo rằng cổng TCP 123 đang mở và nghe.

Cấu hình NTP

NTP có thể thăm dò một số nguồn và chọn ra những ứng viên tốt nhất để sử dụng cho việc thiết lập thời gian của hệ thống. Nếu mất kết nối mạng, NTP sẽ sử dụng các điều chỉnh đã có trước đó từ lịch sử của nó để ước tính các điều chỉnh trong tương lai.

Tùy thuộc vào từng bản phân phối Linux, danh sách máy chủ thời gian mạng sẽ được lưu trữ ở những nơi khác nhau. Hãy giả sử rằng `ntp` đã được cài đặt trên máy.

Tệp `/etc/ntp.conf` chứa thông tin cấu hình về cách hệ thống đồng bộ hóa với thời gian mạng. Tệp này có thể được đọc và sửa đổi bằng cách sử dụng `vi` hoặc `nano`.

Theo mặc định, máy chủ NTP được sử dụng sẽ được chỉ định trong một phần như sau:

```
# Use public servers from the pool.ntp.org project.
# Please consider joining the pool (http://www.pool.ntp.org/join.html).
server 0.centos.pool.ntp.org iburst
server 1.centos.pool.ntp.org iburst
server 2.centos.pool.ntp.org iburst
server 3.centos.pool.ntp.org iburst
```

Cú pháp để thêm máy chủ NTP sẽ như sau:

```
server (IP Address)
server server.url.localhost
```

Địa chỉ máy chủ có thể là địa chỉ IP hoặc URL nếu DNS đã được cấu hình đúng cách. Trong trường hợp này, máy chủ sẽ luôn được truy vấn.

Quản trị viên mạng cũng có thể cân nhắc sử dụng (hoặc thiết lập) *nhóm* (pool). Trong trường hợp này, chúng ta sẽ giả định rằng có nhiều nhà cung cấp NTP đều chạy trình nền NTP và có cùng một mốc thời gian. Khi khách hàng truy vấn một nhóm, nhà cung cấp sẽ được chọn ngẫu nhiên. Điều này giúp phân phối tải mạng giữa nhiều máy để không có máy nào trong nhóm phải xử lý tất cả các truy vấn NTP.

Thông thường, `/etc/ntp.conf` sẽ được điền vào một nhóm máy chủ có tên là `pool.ntp.org` (ví dụ như `server 0.centos.pool.ntp.org` là nhóm NTP mặc định được cung cấp cho các máy CentOS).

pool.ntp.org

Các máy chủ NTP được sử dụng theo mặc định là một dự án mã nguồn mở. Bạn có thể tìm thêm thông tin tại ntppool.org.

Hãy xem xét liệu NTP Pool có phù hợp với mục đích sử dụng của bạn hay không. Nếu doanh nghiệp, tổ chức hay cuộc sống của con người phụ thuộc vào việc chọn đúng thời điểm hoặc có thể bị tổn hại nếu chọn sai thì bạn không nên “gỡ nó ra khỏi internet”. NTP Pool nhìn chung có chất lượng rất cao, nhưng đây là dịch vụ do các tình nguyện viên điều hành trong thời gian rảnh rỗi. Vui lòng trao đổi với nhà cung cấp dịch vụ và thiết bị của bạn về việc thiết lập dịch vụ đáng tin cậy tại địa phương cho bạn. Hãy xem thêm các điều khoản dịch vụ của chúng tôi. Chúng tôi khuyên dùng máy chủ thời gian từ Meinberg nhưng bạn cũng có thể tìm thấy máy chủ thời gian từ End Run, Spectracom và nhiều máy chủ khác.

— ntppool.org

ntpdate

Trong quá trình thiết lập ban đầu, thời gian hệ thống và NTP có thể bất đồng bộ nghiêm trọng. Nếu *phần bù* giữa thời gian hệ thống và NTP lớn hơn 17 phút thì trình nền NTP sẽ không thực hiện thay đổi về thời gian hệ thống. Trong trường hợp này, chúng ta sẽ phải can thiệp theo cách thủ công.

Đầu tiên, nếu `ntpd` đang chạy thì chúng ta cần phải *dừng* nó lại. Hãy sử dụng `systemctl stop ntpd` để làm việc này.

Tiếp theo, hãy sử dụng `ntpdate pool.ntp.org` để thực hiện một phiên đồng bộ hóa đơn ban đầu, trong đó `pool.ntp.org` là địa chỉ IP hoặc URL của máy chủ NTP. Chúng ta có thể sẽ cần nhiều hơn một phiên đồng bộ hóa.

ntpq

`ntpq` là một tiện ích để theo dõi trạng thái của NTP. Khi trình nền NTP đã được khởi động và định cấu hình, `ntpq` có thể được sử dụng để kiểm tra trạng thái của nó:

```
$ ntpq -p
      remote           refid      st t when poll reach  delay  offset  jitter
=====
+37.44.185.42      91.189.94.4    3 u   86  128  377  126.509  -20.398  6.838
+ntp2.0x00.lv     193.204.114.233 2 u   82  128  377  143.885   -8.105  8.478
*inspektor-vlan1 121.131.112.137 2 u   17  128  377  112.878  -23.619  7.959
```

```
b1-66er.matrix. 18.26.4.105      2 u 484 128 10 34.907 -0.811 16.123
```

Trong trường hợp này, `-p` chính là *print* và nó sẽ in một bản tóm tắt về các đơn vị ngang hàng. Địa chỉ máy chủ cũng có thể được trả về dưới dạng địa chỉ IP bằng cách sử dụng `-n`.

remote

Tên máy chủ của nhà cung cấp NTP.

refid

ID tham chiếu của nhà cung cấp NTP.

st

Tầng của nhà cung cấp.

when

Số giây kể từ phiên truy vấn cuối cùng.

poll

Số giây giữa các phiên truy vấn.

reach

ID trạng thái để biết liệu máy chủ có được truy cập hay không. Kết nối thành công sẽ tăng con số này lên 1.

delay

Thời gian tính bằng mili giây giữa phiên truy vấn và phản hồi của máy chủ.

offset

Thời gian tính bằng mili giây giữa thời gian hệ thống và thời gian NTP.

jitter

Phần bù tính bằng mili giây giữa thời gian hệ thống và NTP trong phiên truy vấn cuối cùng.

`ntpq` cũng có chế độ tương tác. Chế độ này sẽ được truy cập khi lệnh được chạy mà không có tùy chọn hoặc đối số. Tùy chọn `?` sẽ trả về danh sách các lệnh mà `ntpq` sẽ nhận ra.

chrony

`chrony` là một cách khác để triển khai NTP. Nó được cài đặt theo mặc định trên một số hệ thống Linux nhưng cũng có sẵn để tải xuống trên tất cả các bản phân phối chính. `chronyd` là trình nền

`chrony` và `chronyc` là giao diện dòng lệnh. Chúng ta có thể sẽ cần phải khởi động và kích hoạt `chronyd` trước khi tương tác với `chronyc`.

Nếu cài đặt `chrony` có cấu hình mặc định thì việc sử dụng lệnh `chronyc tracking` sẽ cung cấp thông tin về NTP và thời gian hệ thống:

```
$ chronyc tracking
Reference ID      : 3265FB3D (bras-vprn-toroon2638w-lp130-11-50-101-251-61.dsl.)
Stratum          : 3
Ref time (UTC)   : Thu Jan 09 19:18:35 2020
System time      : 0.000134029 seconds fast of NTP time
Last offset      : +0.000166506 seconds
RMS offset       : 0.000470712 seconds
Frequency        : 919.818 ppm slow
Residual freq    : +0.078 ppm
Skew             : 0.555 ppm
Root delay       : 0.006151616 seconds
Root dispersion  : 0.010947504 seconds
Update interval  : 129.8 seconds
Leap status      : Normal
```

Kết quả đầu ra này chứa nhiều thông tin hơn đầu ra từ các cách triển khai khác.

Reference ID

ID tham chiếu và tên mà máy tính hiện được đồng bộ hóa.

Stratum

Số bước nhảy ngắn tới máy tính có đồng hồ tham chiếu đính kèm.

Ref time

Đây là thời gian UTC mà phép đo cuối cùng từ nguồn tham chiếu được thực hiện.

System time

Độ trễ đồng hồ hệ thống từ máy chủ được đồng bộ hóa.

Last offset

Phần bù ước tính của lần cập nhật đồng hồ cuối cùng.

RMS offset

Trung bình dài hạn của giá trị phần bù.

Frequency

Đây là tỷ lệ lệch của đồng hồ hệ thống nếu chronyd không sửa nó. Nó được cung cấp theo ppm (parts per million - phần triệu).

Residual freq

Tần số dư biểu thị sự khác biệt giữa các phép đo từ nguồn tham chiếu và tần số hiện đang được sử dụng.

Skew

Giới hạn lỗi ước tính của tần số.

Root delay

Tổng độ trễ đường dẫn mạng tới máy tính tầng mà máy tính đang được đồng bộ hóa từ đó.

Leap status

Đây là trạng thái nhảy vọt có thể có một trong các giá trị sau – bình thường, chèn giây, xóa giây hoặc không được đồng bộ hóa.

Chúng ta cũng có thể xem thông tin chi tiết về bản cập nhật NTP hợp lệ cuối cùng:

```
# chrony ntpdata
Remote address : 172.105.97.111 (AC69616F)
Remote port   : 123
Local address  : 192.168.122.81 (C0A87A51)
Leap status   : Normal
Version       : 4
Mode          : Server
Stratum       : 2
Poll interval  : 6 (64 seconds)
Precision     : -25 (0.000000030 seconds)
Root delay    : 0.000381 seconds
Root dispersion : 0.000092 seconds
Reference ID   : 61B7CE58 ( )
Reference time : Mon Jan 13 21:50:03 2020
Offset        : +0.000491960 seconds
Peer delay    : 0.004312567 seconds
Peer dispersion : 0.000000068 seconds
Response time : 0.000037078 seconds
Jitter asymmetry: +0.00
NTP tests     : 111 111 1111
Interleaved   : No
Authenticated : No
```



```
TX timestamping : Daemon
RX timestamping : Kernel
Total TX       : 15
Total RX       : 15
Total valid RX : 15
```

Cuối cùng, `chronyc sources` sẽ trả về thông tin về máy chủ NTP được sử dụng để đồng bộ hóa thời gian:

```
$ chronyc sources
210 Number of sources = 0
MS Name/IP address      Stratum Poll Reach LastRx Last sample
=====
```

Hiện tại, máy này chưa có các nguồn được cấu hình. Chúng ta có thể thêm nguồn từ `pool.ntp.org` bằng cách mở tệp cấu hình `chrony`. Tệp này thường sẽ được đặt tại `/etc/chrony.conf`. Khi mở tệp này, chúng ta sẽ thấy một số máy chủ được liệt kê theo mặc định:

```
210 Number of sources = 0
MS Name/IP address      Stratum Poll Reach LastRx Last sample
=====
# Most computers using chrony will send measurement requests to one or
# more 'NTP servers'. You will probably find that your Internet Service
# Provider or company have one or more NTP servers that you can specify.
# Failing that, there are a lot of public NTP servers. There is a list
# you can access at http://support.ntp.org/bin/view/Servers/WebHome or
# you can use servers from the 3.arch.pool.ntp.org project.

! server 0.arch.pool.ntp.org iburst iburst
! server 1.arch.pool.ntp.org iburst iburst
! server 2.arch.pool.ntp.org iburst iburst

! pool 3.arch.pool.ntp.org iburst
```

Các máy chủ này cũng sẽ đóng vai trò là một hướng dẫn cú pháp khi truy cập vào máy chủ. Tuy nhiên, trong trường hợp này, chúng ta sẽ chỉ xóa `!` ở đầu mỗi dòng, từ đó bỏ chú thích những dòng này và sử dụng các máy chủ mặc định từ dự án `pool.ntp.org`.

Ngoài ra, trong tệp này, chúng ta có thể chọn thay đổi cấu hình mặc định liên quan đến giới hạn lỗi và phần bù cũng như vị trí của `driftfile` và `keyfile`.

Trên máy này, chúng ta cần thực hiện một phiên chỉnh sửa lớn cho đồng hồ ban đầu. Chúng ta sẽ chọn bỏ chú thích cho dòng sau:

```
! makestep 1.0 3
```

Sau khi thực hiện các thay đổi đối với tệp cấu hình, hãy khởi động lại dịch vụ `chronyd` rồi sử dụng `chronyc makestep` để điều chỉnh đồng hồ hệ thống theo cách thủ công:

```
# chronyc makestep  
200 OK
```

Và sau đó hãy sử dụng `chronyc tracking` như trước để xác minh rằng các thay đổi đã diễn ra.

Bài tập Hướng dẫn

1. Hãy nhập thuật ngữ thích hợp cho mỗi định nghĩa:

Định nghĩa	Thuật ngữ
Một máy tính sẽ chia sẻ thời gian mạng với bạn	
Khoảng cách từ một đồng hồ tham chiếu, tính bằng bước nhảy ngắn hoặc bước nhảy dài	
Sự khác biệt giữa thời gian hệ thống và thời gian mạng	
Sự khác biệt giữa thời gian hệ thống và thời gian mạng kể từ cuộc thăm dò NTP cuối cùng	
Nhóm máy chủ cung cấp thời gian mạng và chia sẻ tải giữa chúng	

2. Hãy chỉ ra lệnh bạn sẽ sử dụng để xuất các giá trị sau:

Giá trị	<code>chronyc tracking</code>	<code>timedatectl show-timesync --all</code>	<code>ntpq -pn</code>	<code>chrony ntpdata</code>	<code>chronyc sources</code>
Phương sai độ trễ					
Độ trôi					
Quãng của phiên thăm dò					
Phần bù					
Tầng					
Địa chỉ IP của nhà cung cấp					
Độ trễ gốc					

3. Bạn đang thiết lập một mạng doanh nghiệp bao gồm một máy chủ Linux và một số máy tính Linux. Máy chủ có địa chỉ IP tĩnh là 192.168.0.101. Bạn quyết định rằng máy chủ sẽ kết nối với

`pool.ntp.org` và sau đó cung cấp thời gian NTP cho máy tính. Hãy mô tả cấu hình của máy chủ và máy tính.

4. Có một máy Linux có thời gian không chính xác. Hãy mô tả các bước bạn sẽ thực hiện để khắc phục sự cố NTP.

Bài tập Mở rộng

1. Hãy tìm hiểu sự khác biệt giữa SNTP và NTP.

SNTP	NTP

2. Tại sao quản trị viên hệ thống có thể chọn *không* sử dụng `pool.ntp.org`?

3. Quản trị viên hệ thống sẽ chọn tham gia hoặc đóng góp cho dự án `pool.ntp.org` bằng cách nào?

Tóm tắt

Trong bài học này, chúng ta đã học về:

- NTP là gì và tại sao nó lại quan trọng.
- Định cấu hình trình nền NTP từ dự án `pool.ntp.org`.
- Sử dụng `ntpq` để xác minh cấu hình NTP.
- Sử dụng `chrony` làm dịch vụ NTP thay thế.

Các lệnh được sử dụng trong bài học này:

`timedatectl show-timesync --all`

Hiển thị thông tin SNTP nếu sử dụng `timedatectl`.

`ntpdate <address>`

Thực hiện cập nhật bước nhảy NTP thủ công một lần.

`ntpq -p`

In lịch sử các cuộc thăm dò gần đây của NTP. `-n` sẽ thay thế URL bằng địa chỉ IP.

`chronyc tracking`

Hiển thị trạng thái NTP nếu sử dụng `chrony`.

`chronyc ntpdata`

Hiển thị thông tin NTP về cuộc thăm dò cuối cùng.

`chronyc sources`

Hiển thị thông tin về các nhà cung cấp NTP.

`chronyc makestep`

Thực hiện cập nhật bước nhảy NTP thủ công một lần nếu sử dụng `chrony`.

Đáp án Bài tập Hướng dẫn

1. Hãy nhập thuật ngữ thích hợp cho mỗi định nghĩa:

Định nghĩa	Thuật ngữ
Một máy tính sẽ chia sẻ thời gian mạng với bạn	Nhà cung cấp
Khoảng cách từ một đồng hồ tham chiếu, tính bằng bước nhảy ngắn hoặc bước nhảy dài	Tầng
Sự khác biệt giữa thời gian hệ thống và thời gian mạng	Phần bù
Sự khác biệt giữa thời gian hệ thống và thời gian mạng kể từ cuộc thăm dò NTP cuối cùng	Jitter
Nhóm máy chủ cung cấp thời gian mạng và chia sẻ tải giữa chúng	Nhóm

2. Hãy chỉ ra lệnh bạn sẽ sử dụng để xuất các giá trị sau:

Giá trị	<code>chronyc tracking</code>	<code>timedatectl show-timesync --all</code>	<code>ntpq -pn</code>	<code>chrony ntpdata</code>	<code>chronyc sources</code>
Phương sai độ trễ		X	X		
Độ trôi					
Quãng của phiên thăm dò	X	X	X (cột when)	X	X
Phần bù	X		X	X	
Tầng	X	X	X	X	X
Địa chỉ IP của nhà cung cấp		X	X	X	X
Độ trễ gốc	X			X	

3. Bạn đang thiết lập một mạng doanh nghiệp bao gồm một máy chủ Linux và một số máy tính Linux. Máy chủ có địa chỉ IP tĩnh là 192.168.0.101. Bạn quyết định rằng máy chủ sẽ kết nối với

`pool.ntp.org` và sau đó cung cấp thời gian NTP cho máy tính. Hãy mô tả cấu hình của máy chủ và máy tính.

Hãy đảm bảo rằng máy chủ có dịch vụ `ntpd` đang chạy chứ không phải `SNTP`. Hãy sử dụng nhóm `pool.ntp.org` trong tệp `/etc/ntp.conf` hoặc `/etc/chrony.conf`. Đối với mỗi máy khách, hãy chỉ định `192.168.0.101` trong mỗi tệp `/etc/ntp.conf` hoặc `/etc/chrony.conf`.

4. Có một máy Linux có thời gian không chính xác. Hãy mô tả các bước bạn sẽ thực hiện để khắc phục sự cố NTP.

Đầu tiên, hãy đảm bảo rằng máy đang được kết nối với Internet. Hãy sử dụng `ping` cho việc này. Sau đó, hãy kiểm tra xem dịch vụ `ntpd` hoặc `SNTP` có đang chạy hay không bằng cách sử dụng `systemctl status ntpd` hoặc `systemctl status systemd-timesyncd`. Bạn có thể thấy thông báo lỗi cung cấp các thông tin hữu ích. Cuối cùng, hãy sử dụng lệnh như `ntpq -p` hoặc `chrony track` để xác minh xem có bất kỳ yêu cầu nào được đưa ra hay không. Nếu thời gian hệ thống khác biệt đáng kể so với thời gian mạng thì có thể thời gian hệ thống đó được coi là "thời gian loạn" và sẽ không bị thay đổi nếu không có sự can thiệp thủ công. Trong trường hợp này, hãy sử dụng lệnh từ bài học trước hoặc `ntpdate pool.ntp.org` để thực hiện đồng bộ hóa ntp một lần.

Đáp án Bài tập Mở rộng

1. Hãy tìm hiểu sự khác biệt giữa SNTP và NTP.

SNTP	NTP
kém chính xác	chính xác hơn
yêu cầu ít tài nguyên hơn	đòi hỏi nhiều tài nguyên hơn
không thể hoạt động như một nhà cung cấp thời gian	có thể hoạt động như một nhà cung cấp thời gian
chỉ thời gian các bước nhảy dài	thời gian các bước nhảy dài hoặc thời gian xoay
yêu cầu thời gian từ một nguồn duy nhất	có thể giám sát nhiều máy chủ NTP và sử dụng nhà cung cấp tối ưu

2. Tại sao quản trị viên hệ thống có thể chọn *không* sử dụng `pool.ntp.org`?

Theo `ntppool.org`: Nếu việc có thời gian chính xác là vô cùng quan trọng, bạn nên xem xét một giải pháp thay thế. Tương tự, nếu nhà cung cấp Internet có máy chủ thời gian, bạn được khuyến nghị nên sử dụng máy chủ đó.

3. Quản trị viên hệ thống sẽ chọn tham gia hoặc đóng góp cho dự án `pool.ntp.org` bằng cách nào?

Theo `www.ntppool.org`: Máy chủ của bạn phải có địa chỉ IP tĩnh và kết nối Internet cố định. Địa chỉ IP tĩnh không được thay đổi chút nào hoặc ít hơn một lần mỗi năm. Ngoài ra, yêu cầu về băng thông rất khiêm tốn: băng thông 384 - 512 Kbit. Máy chủ tầng 3 hoặc 4 đều có thể tham gia.



108.2 Ghi nhật ký Hệ thống

Tham khảo các mục tiêu LPI

[LPIC-1 version 5.0, Exam 102, Objective 108.2](#)

Khối lượng

4

Các lĩnh vực kiến thức chính

- Cấu hình cơ bản của rsyslog.
- Hiểu về cơ sở các tiện ích tiêu chuẩn, các ưu tiên và hành động.
- Truy vấn hành trình systemd.
- Lọc dữ liệu hành trình hệ thống theo các tiêu chí như ngày, dịch vụ hoặc mức độ ưu tiên.
- Định cấu hình lưu trữ hành trình hệ thống liên tục và kích thước hành trình.
- Xóa dữ liệu hành trình hệ thống cũ.
- Truy xuất dữ liệu hành trình systemd từ hệ thống cứu hộ hoặc bản sao hệ thống tệp.
- Hiểu sự tương tác của rsyslog với systemd-journald.
- Cấu hình logrotate.
- Nhận thức về syslog và syslog-ng.

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- `/etc/rsyslog.conf`
- `/var/log/`
- `logger`
- `logrotate`

- `/etc/logrotate.conf`
- `/etc/logrotate.d/`
- `journalctl`
- `systemd-cat`
- `/etc/systemd/journald.conf`
- `/var/log/journal/`



108.1 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	108 Dịch vụ Hệ thống thiết yếu
Mục tiêu:	108.2 Ghi nhật ký Hệ thống
Bài:	1 trên 2

Giới thiệu

Nhật ký có thể được xem là người bạn thân thiết của quản trị viên hệ thống. Nhật ký là các tệp (thường là tệp văn bản) có chứa tất cả các sự kiện mạng và hệ thống được ghi lại theo thứ tự thời gian kể từ thời điểm hệ thống được khởi động. Do đó, phạm vi thông tin có thể tìm thấy trong nhật ký bao gồm hầu hết mọi khía cạnh của hệ thống: các lần xác thực không thành công, lỗi chương trình và dịch vụ, máy chủ bị tường lửa chặn, v.v. Như chúng ta có thể đoán được, nhật ký sẽ khiến công việc của quản trị viên hệ thống trở nên dễ dàng hơn rất nhiều trong các trường hợp cần xử lý sự cố, kiểm tra tài nguyên, phát hiện hành vi bất thường của chương trình, v.v.

Trong bài học này, chúng ta sẽ thảo luận về một trong những công cụ hỗ trợ ghi nhật ký phổ biến nhất hiện có trong các bản phân phối GNU/Linux: `rsyslog`. Chúng ta sẽ nghiên cứu các loại nhật ký khác nhau hiện có, nơi chúng được lưu trữ, các thông tin có trong nhật ký cũng như cách lấy và lọc các thông tin đó. Chúng ta cũng sẽ thảo luận về cách lưu giữ nhật ký trong các máy chủ tập trung trên mạng IP, công cụ xoay vòng nhật ký và bộ đệm vòng hạt nhân.

Ghi nhật ký Hệ thống

Thời điểm hạt nhân và các tiến trình khác nhau trong hệ thống bắt đầu chạy và liên lạc với nhau,

rất nhiều thông tin sẽ được tạo ra dưới dạng thông báo và — phần lớn trong số chúng — sẽ được gửi tới nhật ký.

Nếu không ghi nhật ký, việc tìm kiếm một sự kiện xảy ra trên máy chủ sẽ khiến quản trị viên hệ thống phải đau đầu. Từ đó chúng ta có thể thấy được tầm quan trọng của việc xây dựng một phương thức theo dõi tập trung và tiêu chuẩn hóa mọi sự kiện trên hệ thống. Nhật ký là yếu tố quyết định và sẽ cung cấp các thông tin cần thiết trong việc xử lý sự cố và bảo mật, đồng thời cũng là nguồn dữ liệu đáng tin cậy để hiểu được số liệu thống kê hệ thống và đưa ra các dự đoán xu hướng chính xác.

Bỏ qua `systemd-journald` (mà chúng ta sẽ thảo luận trong bài học tiếp theo), việc ghi nhật ký theo cách truyền thống được xử lý bởi ba dịch vụ chuyên dụng chính: `syslog`, `syslog-ng` (`syslog` thế hệ mới) và `rsyslog` (“hệ thống xử lý ghi nhật ký siêu nhanh”). `rsyslog` đã mang đến những cải tiến quan trọng (chẳng hạn như hỗ trợ RELP) và đã trở thành lựa chọn phổ biến nhất hiện nay. Mỗi dịch vụ này đều sẽ thu thập thông báo từ các dịch vụ và chương trình khác rồi lưu trữ chúng trong tệp nhật ký, thường là trong `/var/log`. Tuy nhiên, một số dịch vụ sẽ tự xử lý nhật ký của chúng (ví dụ như máy chủ web Apache HTTPD hoặc hệ thống in CUPS). Tương tự, nhân Linux sẽ sử dụng bộ đệm vòng trong bộ nhớ để lưu trữ các thông báo nhật ký của chính nó.

NOTE

RELP (Reliable Event Logging Protocol) là viết tắt của *Giao thức ghi nhật ký sự kiện đáng tin cậy* và sẽ mở rộng chức năng của giao thức `syslog` để cung cấp chức năng gửi thông báo đáng tin cậy.

Vì `rsyslog` đã trở thành công cụ ghi nhật ký tiêu chuẩn trong tất cả các bản phân phối chính nên chúng ta sẽ tập trung vào nó trong bài học này. `rsyslog` sử dụng mô hình máy khách-máy chủ (client-server). Máy khách và máy chủ có thể hoạt động trên cùng một máy chủ (host) hoặc trong các máy khác nhau. Thông báo sẽ được gửi và nhận ở một định dạng cụ thể và có thể được lưu giữ trong các máy chủ `rsyslog` tập trung trên các mạng IP. Trình nền của `rsyslog` — `rsyslogd` — hoạt động cùng với `klogd` (quản lý các thông báo hạt nhân). Trong các phần tiếp theo, chúng ta sẽ thảo luận về `rsyslog` và các thành phần cấu thành nên chức năng ghi nhật ký của nó.

NOTE

Trình nền (Daemon) là một dịch vụ chạy ngầm. Hãy lưu ý ký tự `d` đứng cuối trong tên của trình nền: `klogd` hoặc `rsyslogd`.

Các loại Nhật ký

Vì nhật ký là dữ liệu *biến* nên chúng thường được tìm thấy trong `/var/log`. Nói một cách đại khái, chúng có thể được phân loại thành *nhật ký hệ thống* và *nhật ký dịch vụ hoặc chương trình*.

Hãy cùng xem một số nhật ký hệ thống và các thông tin mà chúng lưu giữ:

/var/log/auth.log

Các hoạt động liên quan đến quá trình xác thực: người dùng đã đăng nhập, thông tin sudo, các công việc định kỳ, lần đăng nhập không thành công, v.v.

/var/log/syslog

Một tệp tập trung dành cho hầu hết tất cả các nhật ký được ghi lại bởi rsyslogd. Vì nó bao gồm rất nhiều thông tin nên nhật ký sẽ được phân phối trên các tệp khác theo cấu hình được cung cấp trong /etc/rsyslog.conf.

/var/log/debug

Thông tin gỡ lỗi từ các chương trình.

/var/log/kern.log

Thông báo từ hạt nhân.

/var/log/messages

Các thông báo mang tính thông tin liên quan đến các dịch vụ khác mà không liên quan đến hạt nhân. Đây cũng là đích đến nhật ký máy khách từ xa mặc định trong một phiên triển khai nhật ký máy chủ tập trung.

/var/log/daemon.log

Thông tin liên quan đến trình nền hoặc các dịch vụ chạy ngầm.

/var/log/mail.log

Thông tin liên quan đến máy chủ email (ví dụ như postfix).

/var/log/Xorg.0.log

Thông tin liên quan đến thẻ đồ họa.

/var/run/utmp và /var/log/wtmp

Các phiên đăng nhập thành công.

/var/log/btmp

Các phiên đăng nhập không thành công (ví dụ như việc tấn công brute force thông qua ssh).

/var/log/faillog

Các phiên xác thực không thành công.

/var/log/lastlog

Ngày và giờ đăng nhập gần đây của người dùng.

Bây giờ, hãy cùng xem một vài ví dụ về nhật ký dịch vụ:

/var/log/cups/

Thư mục nhật ký của *Hệ thống in Unix phổ thông*. Nó thường bao gồm các tệp nhật ký mặc định sau: `error_log`, `page_log` và `access_log`.

/var/log/apache2/ hoặc /var/log/httpd

Thư mục nhật ký của *Máy chủ web Apache*. Nó thường bao gồm các tệp nhật ký mặc định sau: `access.log`, `error_log` và `other_vhosts_access.log`.

/var/log/mysql

Thư mục nhật ký của *Hệ thống quản lý cơ sở dữ liệu quan hệ MySQL*. Nó thường bao gồm các tệp nhật ký mặc định sau: `error_log`, `mysql.log` và `mysql-slow.log`.

/var/log/samba/

Thư mục nhật ký của giao thức *Khối Thông báo Phiên* (SMB). Nó thường bao gồm các tệp nhật ký mặc định sau: `log.`, `log.nmbd` và `log.smbd`.

NOTE

Tên và nội dung chính xác của tệp nhật ký có thể sẽ khác nhau tùy vào bản phân phối Linux. Ngoài ra còn có các nhật ký dành riêng cho các bản phân phối cụ thể (chẳng hạn như `/var/log/dpkg.log` (chứa thông tin liên quan đến các gói dpkg) trong Debian GNU/Linux và các biến thể của nó).

Đọc Nhật ký

Để đọc tệp nhật ký, trước tiên hãy đăng nhập dưới tên siêu người dùng hoặc người dùng có quyền đọc tệp. Chúng ta có thể sử dụng nhiều tiện ích đa dạng như:

less hoặc more

Các tùy chọn trang cho phép xem và cuộn từng trang một:

```
root@debian:~# less /var/log/auth.log
Sep 12 18:47:56 debian sshd[441]: Received SIGHUP; restarting.
Sep 12 18:47:56 debian sshd[441]: Server listening on 0.0.0.0 port 22.
Sep 12 18:47:56 debian sshd[441]: Server listening on :: port 22.
Sep 12 18:47:56 debian sshd[441]: Received SIGHUP; restarting.
Sep 12 18:47:56 debian sshd[441]: Server listening on 0.0.0.0 port 22.
Sep 12 18:47:56 debian sshd[441]: Server listening on :: port 22.
Sep 12 18:49:46 debian sshd[905]: Accepted password for carol from 192.168.1.65 port
44296 ssh2
Sep 12 18:49:46 debian sshd[905]: pam_unix(sshd:session): session opened for user carol
```

```
by (uid=0)
Sep 12 18:49:46 debian systemd-logind[331]: New session 2 of user carol.
Sep 12 18:49:46 debian systemd: pam_unix(systemd-user:session): session opened for user
carol by (uid=0)
(...)
```

zless hoặc zmore

Tương tự như `less` và `more` nhưng sử dụng cho nhật ký được nén bằng `gzip` (một chức năng phổ biến của `logrotate`):

```
root@debian:~# zless /var/log/auth.log.3.gz
Aug 19 20:05:57 debian sudo:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/sbin/shutdown -h now
Aug 19 20:05:57 debian sudo: pam_unix(sudo:session): session opened for user root by
carol(uid=0)
Aug 19 20:05:57 debian lightdm: pam_unix(lightdm-greeter:session): session closed for
user lightdm
Aug 19 23:50:49 debian systemd-logind[333]: Watching system buttons on /dev/input/event2
(Power Button)
Aug 19 23:50:49 debian systemd-logind[333]: Watching system buttons on /dev/input/event3
(Sleep Button)
Aug 19 23:50:49 debian systemd-logind[333]: Watching system buttons on /dev/input/event4
(Video Bus)
Aug 19 23:50:49 debian systemd-logind[333]: New seat seat0.
Aug 19 23:50:49 debian sshd[409]: Server listening on 0.0.0.0 port 22.
(...)
```

tail

Xem những dòng cuối cùng trong một tệp (mặc định là 10 dòng). Sức mạnh của `tail` — trong một phạm vi rộng hơn — nằm trong khoá chuyển `-f`; với khoá chuyển này, nó sẽ tự động hiển thị các dòng mới khi chúng được thêm vào:

```
root@suse-server:~# tail -f /var/log/messages
2019-09-14T13:57:28.962780+02:00 suse-server sudo: pam_unix(sudo:session): session closed
for user root
2019-09-14T13:57:38.038298+02:00 suse-server sudo:    carol : TTY=pts/0 ; PWD=/home/carol
; USER=root ; COMMAND=/usr/bin/tail -f /var/log/messages
2019-09-14T13:57:38.039927+02:00 suse-server sudo: pam_unix(sudo:session): session opened
for user root by carol(uid=0)
2019-09-14T14:07:22+02:00 debian carol: appending new message from client to remote
server...
```


head

Xem dòng đầu tiên trong tệp (mặc định là 10 dòng):

```

root@suse-server:~# head -5 /var/log/mail
2019-06-29T11:47:59.219806+02:00 suse-server postfix/postfix-script[1732]: the Postfix
mail system is not running
2019-06-29T11:48:01.355361+02:00 suse-server postfix/postfix-script[1925]: starting the
Postfix mail system
2019-06-29T11:48:01.391128+02:00 suse-server postfix/master[1930]: daemon started --
version 3.3.1, configuration /etc/postfix
2019-06-29T11:55:39.247462+02:00 suse-server postfix/postfix-script[3364]: stopping the
Postfix mail system
2019-06-29T11:55:39.249375+02:00 suse-server postfix/master[1930]: terminating on signal
15

```

grep

Tiện ích lọc cho phép người dùng tìm kiếm các chuỗi cụ thể:

```

root@debian:~# grep "dhclient" /var/log/syslog
Sep 13 11:58:48 debian dhclient[448]: DHCPREQUEST of 192.168.1.4 on enp0s3 to 192.168.1.1
port 67
Sep 13 11:58:49 debian dhclient[448]: DHCPACK of 192.168.1.4 from 192.168.1.1
Sep 13 11:58:49 debian dhclient[448]: bound to 192.168.1.4 -- renewal in 1368 seconds.
(...)

```

Như có thể thấy được, kết quả đầu ra được in theo định dạng sau:

- Dấu thời gian
- Tên máy chủ nơi gửi thông báo
- Tên chương trình/dịch vụ tạo ra thông báo
- PID của chương trình tạo ra thông báo
- Diễn tả hành động đã xảy ra

Có một số ví dụ mà trong đó nhật ký không phải là văn bản mà là các tệp nhị phân và — do đó — chúng ta phải sử dụng các lệnh đặc biệt để phân tích chúng:

/var/log/wtmp

Sử dụng `who` (hoặc `w`):

```
root@debian:~# who
root    pts/0      2020-09-14 13:05 (192.168.1.75)
root    pts/1      2020-09-14 13:43 (192.168.1.75)
```

/var/log/btmp

Sử dụng `utmpdump` hoặc `last -f`:

```
root@debian:~# utmpdump /var/log/btmp
Utmp dump of /var/log/btmp
[6] [01287] [  ] [dave    ] [ssh:notty  ] [192.168.1.75      ] [192.168.1.75    ]
[2019-09-07T19:33:32,000000+0000]
```

/var/log/faillog

Sử dụng `faillog`:

```
root@debian:~# faillog -a | less
Login      Failures Maximum Latest           On

root       0         0   01/01/70 01:00:00 +0100
daemon    0         0   01/01/70 01:00:00 +0100
bin        0         0   01/01/70 01:00:00 +0100
sys        0         0   01/01/70 01:00:00 +0100
sync      0         0   01/01/70 01:00:00 +0100
games     0         0   01/01/70 01:00:00 +0100
man        0         0   01/01/70 01:00:00 +0100
lp         0         0   01/01/70 01:00:00 +0100
mail      0         0   01/01/70 01:00:00 +0100
(...)
```

/var/log/lastlog

Sử dụng `lastlog`:

```
root@debian:~# lastlog | less
Username      Port      From           Latest

root          Never logged in
daemon       Never logged in
bin          Never logged in
sys          Never logged in
(...)
```

```
sync          Never logged in
```

```

avahi                Never logged in
colord               Never logged in
saned                Never logged in
hplip                Never logged in
carol                pts/1    192.168.1.75  Sat Sep 14 13:43:06 +0200 2019
dave                 pts/3    192.168.1.75  Mon Sep  2 14:22:08 +0200 2019

```

NOTE

Ngoài ra còn có các công cụ đồ họa để đọc tệp nhật ký (ví dụ như `gnome-logs` và `KSystemLog`).

Cách chuyển Thông báo thành Nhật ký

Quá trình sau đây minh họa cách một thông báo được ghi vào tệp nhật ký:

1. Các ứng dụng, dịch vụ và hạt nhân sẽ ghi thông báo vào các tệp đặc biệt (ổ nổi và bộ nhớ đệm), ví dụ như `/dev/log` hoặc `/dev/kmsg`.
2. `rsyslogd` lấy thông tin từ ổ nổi hoặc bộ nhớ đệm.
3. Tùy thuộc vào các quy tắc được tìm thấy trong `/etc/rsyslog.conf` và/hoặc các tệp trong `/etc/rsyslog.d/`, `rsyslogd` sẽ chuyển thông tin đến tệp nhật ký tương ứng (thường được tìm thấy trong `/var/log`).

NOTE

Ổ nổi là một tệp đặc biệt được sử dụng để truyền thông tin giữa các tiến trình khác nhau. Để liệt kê tất cả các ổ nổi trên hệ thống, chúng ta có thể sử dụng lệnh `systemctl list-sockets --all`.

Tiện ích đại diện, Mức độ ưu tiên và Hành động

Tệp cấu hình `rsyslog` là `/etc/rsyslog.conf` (trong một số bản phân phối, chúng ta cũng có thể tìm thấy các tệp cấu hình trong `/etc/rsyslog.d/`). Nó thường được chia thành ba phần: `MODULES` (MÔ-ĐUN), `GLOBAL DIRECTIVES` (CHỈ THỊ TOÀN CỤC) và `RULES` (QUY TẮC). Hãy cùng xem các phần này bằng cách khám phá tệp `rsyslog.conf` trong máy chủ Debian GNU/Linux 10 (buster) — chúng ta cũng có thể sử dụng `sudo less /etc/rsyslog.conf` để thực hiện việc này.

`MODULES` bao gồm các hỗ trợ mô-đun dành cho việc ghi nhật ký, năng lực thông báo và nhận nhật ký UDP/TCP:

```

#####
#### MODULES ####
#####

module(load="imuxsock") # provides support for local system logging

```

```

module(load="imklog") # provides kernel logging support
#module(load="imark") # provides --MARK-- message capability

# provides UDP syslog reception
#module(load="imudp")
#input(type="imudp" port="514")

# provides TCP syslog reception
#module(load="imtcp")
#input(type="imtcp" port="514")

```

GLOBAL DIRECTIVES cho phép chúng ta định cấu hình một số thứ như nhật ký và quyền đối với thư mục nhật ký:

```

#####
#### GLOBAL DIRECTIVES ####
#####

#
# Use traditional timestamp format.
# To enable high precision timestamps, comment out the following line.
#
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat

#
# Set the default permissions for all log files.
#
$FileOwner root
$FileGroup adm
$FileCreateMode 0640
$DirCreateMode 0755
$Umask 0022

#
# Where to place spool and state files
#
$WorkDirectory /var/spool/rsyslog

#
# Include all config files in /etc/rsyslog.d/
#
$IncludeConfig /etc/rsyslog.d/*.conf

```

RULES là nơi chứa *các tiện ích đại diện, mức độ ưu tiên và hành động*. Các cài đặt trong phần này yêu cầu trình nền ghi nhật ký lọc các thông báo theo những quy tắc nhất định và ghi nhật ký hoặc gửi chúng khi cần thiết. Để hiểu các quy tắc này, trước tiên chúng ta cần giải thích các khái niệm về tiện ích đại diện và mức độ ưu tiên của `rsyslog`. Mỗi thông báo nhật ký sẽ được cung cấp một số *tiện ích đại diện* và từ khóa được liên kết với hệ thống phụ nội bộ Linux tạo ra thông báo:

Số	Từ khóa	Mô tả
0	kern	Thông báo nhân Linux
1	user	Thông báo cấp người dùng
2	mail	Hệ thống thư
3	daemon	Trình nền hệ thống
4	auth, authpriv	Thông báo bảo mật/xác thực
5	syslog	Thông báo syslogd
6	lpr	Hệ thống phụ in dòng
7	news	Hệ thống phụ tin tức mạng
8	uucp	Hệ thống phụ UUCP (Giao thức sao chép Unix-sang-Unix)
9	cron	Trình nền đồng hồ
10	auth, authpriv	Thông báo bảo mật/xác thực
11	ftp	Trình nền FTP (Giao thức truyền tệp)
12	ntp	Trình nền NTP (Giao thức thời gian mạng)
13	security	Kiểm tra nhật ký
14	console	Cảnh báo nhật ký
15	cron	Trình nền đồng hồ
16 - 23	local0 đến local7	Sử dụng cục bộ 0 - 7

Thêm vào đó, mỗi thông báo sẽ được gán một mức *ưu tiên*:

Mã	Mức độ nghiêm trọng	Từ khóa	Mô tả
0	Khẩn cấp (emergency)	emerg, panic	Hệ thống không thể sử dụng được

Mã	Mức độ nghiêm trọng	Từ khóa	Mô tả
1	Cảnh báo (alert)	alert	Phải có hành động ngay lập tức
2	Nghiêm trọng (critical)	crit	Tình trạng nghiêm trọng
3	Lỗi (error)	err, error	Tình trạng lỗi
4	Cảnh báo (warning)	warn, warning	Tình trạng cảnh báo
5	Thông báo (notice)	Notice	Tình trạng bình thường nhưng đáng chú ý
6	Thông tin (informational)	info	Thông báo mang tính thông tin
7	Gỡ lỗi (debug)	debug	Thông báo cấp độ gỡ lỗi

Sau đây là một đoạn trích `rsyslog.conf` từ hệ thống Debian GNU/Linux 10 (buster) bao gồm một số quy tắc mẫu:

```
#####
#### RULES ####
#####

# First some standard log files.  Log by facility.
#
auth,authpriv.*      /var/log/auth.log
*.*;auth,authpriv.none  -/var/log/syslog
#cron.*              /var/log/cron.log
daemon.*             -/var/log/daemon.log
kern.*               -/var/log/kern.log
lpr.*                -/var/log/lpr.log
mail.*               -/var/log/mail.log
user.*               -/var/log/user.log

#
# Logging for the mail system.  Split it up so that
# it is easy to write scripts to parse these files.
#
mail.info            -/var/log/mail.info
mail.warn            -/var/log/mail.warn
mail.err             /var/log/mail.err
```

```
#
# Some "catch-all" log files.
#
*.=debug;\
    auth,authpriv.none;\
    news.none;mail.none    -/var/log/debug
*.=info;*.=notice;*.=warn;\
    auth,authpriv.none;\
    cron,daemon.none;\
    mail,news.none        -/var/log/messages
```

Định dạng quy tắc sẽ như sau: <facility>.<priority> <action>

Bộ chọn <facility>.<priority> sẽ lọc các thông báo để khớp. Các mức độ ưu tiên được tính theo thứ bậc, có nghĩa là rsyslog sẽ khớp với các thông báo có mức độ ưu tiên được chỉ định hoặc cao hơn. <action> sẽ hiển thị hành động cần thực hiện (nơi gửi thông báo nhật ký). Dưới đây là một vài ví dụ:

```
auth,authpriv.*          /var/log/auth.log
```

Bất kể mức độ ưu tiên của chúng (*) là gì, tất cả các thông báo từ tiện ích đại diện auth hoặc authpriv đều sẽ được gửi đến /var/log/auth.log.

```
*.*;auth,authpriv.none  -/var/log/syslog
```

Tất cả các thông báo — bất kể mức độ ưu tiên của chúng (*) là gì — từ tất cả các tiện ích đại diện (*) — loại bỏ các thông báo từ auth hoặc authpriv (do đó có hậu tố .none) — đều sẽ được ghi vào /var/log/syslog (dấu trừ (-) trước đường dẫn sẽ ngăn chặn việc ghi đĩa quá mức). Hãy lưu ý dấu chấm phẩy (;) được dùng để phân tách bộ chọn và dấu phẩy (,) được dùng để nối hai tiện ích đại diện trong cùng một quy tắc (auth, authpriv).

```
mail.err                 /var/log/mail.err
```

Thông báo từ tiện ích đại diện mail có mức độ ưu tiên là error hoặc cao hơn (critical, alert hoặc emergency) sẽ được gửi đến /var/log/mail.err.

```
*.=debug;\
    auth,authpriv.none;\
    news.none;mail.none    -/var/log/debug
```

Thông báo từ tất cả các tiện ích đại diện có mức độ ưu tiên debug và không có mức độ (=) nào khác sẽ được ghi vào `/var/log/debug` — ngoại trừ mọi thông báo đến từ tiện ích đại diện `auth`, `authpriv`, `news` và `mail` (lưu ý cú pháp: `;` `\`).

Nhập thủ công vào Nhật ký hệ thống: `logger`

Lệnh `logger` rất hữu ích cho việc tạo tệp lệnh vỏ hoặc cho mục đích thử nghiệm. `logger` sẽ nối thêm bất kỳ thông báo nào nó nhận được vào `/var/log/syslog` (hoặc vào `/var/log/messages` khi đăng nhập vào máy chủ nhật ký trung tâm từ xa như chúng ta sẽ thấy ở phần sau của bài học này):

```
carol@debian:~$ logger this comment goes into "/var/log/syslog"
```

Để in dòng cuối cùng trong `/var/log/syslog`, hãy sử dụng lệnh `tail` với tùy chọn `-1`:

```
root@debian:~# tail -1 /var/log/syslog
Sep 17 17:55:33 debian carol: this comment goes into /var/log/syslog
```

`rsyslog` làm Máy chủ nhật ký trung tâm

Để giải thích vấn đề này, chúng ta sẽ cùng thêm một máy chủ mới vào thiết lập. Bố cục của chúng sẽ như sau:

Vai trò	Tên máy chủ	Hệ điều hành	Địa chỉ IP
Máy chủ nhật ký trung tâm	<code>suse-server</code>	openSUSE Leap 15.1	192.168.1.6
Máy khách	<code>debian</code>	Debian GNU/Linux 10 (buster)	192.168.1.4

Chúng ta hãy bắt đầu bằng việc cấu hình máy chủ. Trước hết, chúng ta phải đảm bảo rằng `rsyslog` đã và đang hoạt động:

```
root@suse-server:~# systemctl status rsyslog
rsyslog.service - System Logging Service
Loaded: loaded (/usr/lib/systemd/system/rsyslog.service; enabled; vendor preset: enabled)
Active: active (running) since Thu 2019-09-17 18:45:58 CEST; 7min ago
Docs: man:rsyslogd(8)
      http://www.rsyslog.com/doc/
Main PID: 832 (rsyslogd)
```



```
Tasks: 5 (limit: 4915)
CGroup: /system.slice/rsyslog.service
└─832 /usr/sbin/rsyslogd -n -iNONE
```

openSUSE sẽ cung cấp tệp cấu hình chuyên dụng để ghi nhật ký từ xa: `/etc/rsyslog.d/remote.conf`. Hãy cùng kích hoạt tính năng nhận thông báo từ máy khách (các máy chủ từ xa) qua TCP. Chúng ta sẽ phải bỏ chú thích cho các dòng tải mô-đun và khởi động máy chủ TCP trên cổng 514:

```
# ##### Receiving Messages from Remote Hosts #####
# TCP Syslog Server:
# provides TCP syslog reception and GSS-API (if compiled to support it)
$ModLoad imtcp.so # load module
##$UDPServerAddress 10.10.0.1 # force to listen on this IP only
$InputTCPServerRun 514 # Starts a TCP server on selected port

# UDP Syslog Server:
#$ModLoad imudp.so # provides UDP syslog reception
##$UDPServerAddress 10.10.0.1 # force to listen on this IP only
#$UDPServerRun 514 # start a UDP syslog server at standard port 514
```

Sau khi hoàn tất, chúng ta sẽ phải khởi động lại dịch vụ `rsyslog` và kiểm tra xem máy chủ có đang nghe trên cổng 514 hay không:

```
root@suse-server:~# systemctl restart rsyslog
root@suse-server:~# netstat -nlt | grep 514
[sudo] password for root:
tcp        0      0 0.0.0.0:514          0.0.0.0:*          LISTEN
2263/rsyslogd
tcp6       0      0 :::514              :::*                LISTEN
2263/rsyslogd
```

Tiếp theo, chúng ta nên mở các cổng trong tường lửa và tải lại cấu hình:

```
root@suse-server:~# firewall-cmd --permanent --add-port 514/tcp
success
root@suse-server:~# firewall-cmd --reload
success
```

NOTE

Với sự xuất hiện của openSUSE Leap 15.0, `firewalld` đã thay thế hoàn toàn

SuSEFirewall12 cổ điển.

Khuôn mẫu và Điều kiện lọc

Theo mặc định, nhật ký của máy khách sẽ được ghi vào tệp `/var/log/messages` của máy chủ — cùng với nhật ký của chính máy chủ. Tuy nhiên, chúng ta sẽ tạo một *khuôn mẫu* và một *điều kiện lọc* để lưu trữ nhật ký của máy khách trong các thư mục rõ ràng của riêng chúng. Để làm như vậy, chúng ta sẽ thêm phần sau vào `/etc/rsyslog.conf` (hoặc `/etc/rsyslog.d/remote.conf`):

```
$template RemoteLogs, "/var/log/remotehosts/%HOSTNAME%/%$NOW%.%syslogseverity-text%.log"
if $FROMHOST-IP=='192.168.1.4' then ?RemoteLogs
& stop
```

Khuôn mẫu

Khuôn mẫu tương ứng với dòng đầu tiên và cho phép chúng ta chỉ định định dạng cho tên nhật ký bằng cách tạo tên tệp động. Một khuôn mẫu sẽ bao gồm:

- Chỉ thị khuôn mẫu (`$template`)
- Tên khuôn mẫu (`RemoteLogs`)
- Văn bản khuôn mẫu ("`/var/log/remotehosts/%HOSTNAME%/%$NOW%.%syslogseverity-text%.log`")
- Các tùy chọn (tùy chọn)

Khuôn mẫu của chúng ta được gọi là `RemoteLogs` và văn bản của nó bao gồm một đường dẫn trong `/var/log`. Tất cả các nhật ký của máy chủ từ xa sẽ đi vào thư mục `remotehosts`. Ở đây, thư mục con sẽ được tạo dựa trên tên máy chủ của máy (`%HOSTNAME%`). Mỗi tên tệp trong thư mục này sẽ bao gồm ngày (`%$NOW%`), mức độ nghiêm trọng (còn gọi là mức độ ưu tiên) của thông báo ở định dạng văn bản (`%syslogseverity-text%`) và hậu tố `.log`. Các từ nằm giữa các dấu phần trăm là *các đặc tính* và cho phép chúng ta truy cập vào nội dung của thông báo nhật ký (ngày, mức độ ưu tiên, v.v.). Thông báo syslog có một số đặc tính được xác định rõ ràng có thể được sử dụng trong các khuôn mẫu. Các đặc tính này được truy cập — và có thể được sửa đổi — bởi *phần thay thế đặc tính* với ngữ ý là đặt chúng giữa các dấu phần trăm.

Điều kiện lọc

Hai dòng còn lại tương ứng với điều kiện lọc và hành động liên quan của nó:

- Bộ lọc dựa trên biểu thức (`if $FROMHOST-IP=='192.168.1.4'`)
- Hành động (`then ?RemoteLogs, & stop`)

Dòng đầu tiên sẽ kiểm tra địa chỉ IP của máy chủ từ xa đang gửi nhật ký và — nếu nó tương đương với địa chỉ của máy khách Debian của chúng ta — nó sẽ áp dụng khuôn mẫu RemoteLogs. Dòng cuối cùng (& stop) sẽ đảm bảo rằng các thông báo sẽ không được gửi đồng thời đến /var/log/messages (mà chỉ gửi đến các tệp trong thư mục /var/log/remotehosts).

NOTE Để tìm hiểu thêm về các mẫu, đặc tính và quy tắc, bạn có thể tham khảo trang hướng dẫn về rsyslog.conf.

Với cấu hình đã được cập nhật, chúng ta sẽ khởi động lại rsyslog và xác nhận rằng chưa có thư mục remotehosts nào trong /var/log:

```
root@suse-server:~# systemctl restart rsyslog
root@suse-server:~# ls /var/log/
acpid                chrony                localmessages        pbl.log              Xorg.0.log
alternatives.log     cups                 mail                 pk_backend_zypp     Xorg.0.log.old
apparmor             firebird             mail.err             samba                YaST2
audit                firewall             mail.info            snapper.log          zypp
boot.log             firewalld            mail.warn            tallylog             zypper.log
boot.msg             krb5                 messages             tuned
boot.omsg            lastlog              mysql                warn
btmp                 lightdm              NetworkManager       wtmp
```

Máy chủ hiện đã được cấu hình. Tiếp theo, chúng ta sẽ cấu hình máy khách.

Một lần nữa, chúng ta phải đảm bảo rằng rsyslog đã được cài đặt và đang chạy:

```
root@debian:~# sudo systemctl status rsyslog
rsyslog.service - System Logging Service
   Loaded: loaded (/lib/systemd/system/rsyslog.service; enabled; vendor preset:
   Active: active (running) since Thu 2019-09-17 18:47:54 CEST; 7min ago
     Docs: man:rsyslogd(8)
           http://www.rsyslog.com/doc/
   Main PID: 351 (rsyslogd)
     Tasks: 4 (limit: 4915)
    CGroup: /system.slice/rsyslog.service
            └─351 /usr/sbin/rsyslogd -n
```

Trong môi trường mẫu, chúng ta đã triển khai phân giải tên trên máy khách bằng cách thêm dòng 192.168.1.6 suse-server vào /etc/hosts. Do đó, chúng ta có thể gọi máy chủ theo tên (suse-server) hoặc địa chỉ IP (192.168.1.6).

Máy khách Debian của chúng ta không có tệp remote.conf trong /etc/rsyslog.d/; vì vậy,

chúng ta sẽ áp dụng các cấu hình của mình trong `/etc/rsyslog.conf`. Chúng ta sẽ viết dòng sau vào cuối tệp:

```
*.* @@suse-server:514
```

Cuối cùng, chúng ta sẽ khởi động lại `rsyslog`.

```
root@debian:~# systemctl restart rsyslog
```

Bây giờ, hãy cùng quay lại máy `suse-server` và kiểm tra sự tồn tại của `remotehosts` trong `/var/log`:

```
root@suse-server:~# ls /var/log/remotehosts/debian/  
2019-09-17.info.log 2019-09-17.notice.log
```

Chúng ta đã có hai nhật ký bên trong `/var/log/remotehosts` như được mô tả trong khuôn mẫu. Để hoàn thành phần này, chúng ta sẽ chạy `tail -f 2019-09-17.notice.log` trên `suse-server` trong khi gửi nhật ký *theo cách thủ công* từ máy khách Debian và xác nhận rằng các thông báo đã được thêm vào tệp nhật ký như dự kiến (tùy chọn `-t` sẽ cung cấp một thẻ cho thông báo):

```
root@suse-server:~# tail -f /var/log/remotehosts/debian/2019-09-17.notice.log  
2019-09-17T20:57:42+02:00 debian dbus[323]: [system] Successfully activated service  
'org.freedesktop.nm_dispatcher'  
2019-09-17T21:01:41+02:00 debian anacron[1766]: Anacron 2.3 started on 2019-09-17  
2019-09-17T21:01:41+02:00 debian anacron[1766]: Normal exit (0 jobs run)
```

```
carol@debian:~$ logger -t DEBIAN-CLIENT Hi from 192.168.1.4
```

```
root@suse-server:~# tail -f /var/log/remotehosts/debian/2019-09-17.notice.log  
2019-09-17T20:57:42+02:00 debian dbus[323]: [system] Successfully activated service  
'org.freedesktop.nm_dispatcher'  
2019-09-17T21:01:41+02:00 debian anacron[1766]: Anacron 2.3 started on 2019-09-17  
2019-09-17T21:01:41+02:00 debian anacron[1766]: Normal exit (0 jobs run)  
2019-09-17T21:04:21+02:00 debian DEBIAN-CLIENT: Hi from 192.168.1.4
```

Cơ chế xoay vòng Nhật ký

Nhật ký sẽ được xoay vòng thường xuyên nhằm phục vụ hai mục đích chính:

- Ngăn các tệp nhật ký cũ sử dụng nhiều dung lượng đĩa hơn mức cần thiết.
- Giữ nhật ký ở độ dài có thể quản lý được để dễ dàng tham khảo.

Tiện ích chịu trách nhiệm xoay vòng nhật ký là `logrotate` và công việc của nó bao gồm các hành vi như di chuyển tệp nhật ký sang một tên mới, lưu trữ và/hoặc nén chúng, đôi khi gửi chúng qua email cho quản trị viên hệ thống và cuối cùng là xóa khi chúng đã cũ đi. Có nhiều quy ước khác nhau để đặt tên cho các tệp nhật ký được xoay vòng (ví dụ: thêm hậu tố có ngày vào tên tệp); tuy nhiên, việc thêm hậu tố có số nguyên là một cách làm phổ biến:

```
root@debian:~# ls /var/log/messages*
/var/log/messages /var/log/messages.1 /var/log/messages.2.gz /var/log/messages.3.gz
/var/log/messages.4.gz
```

Bây giờ, hãy cùng giải thích điều gì sẽ xảy ra trong phiên xoay vòng nhật ký tiếp theo:

1. `messages.4.gz` sẽ bị xóa và sẽ mất đi.
2. Nội dung của `messages.3.gz` sẽ được chuyển sang `messages.4.gz`.
3. Nội dung của `messages.2.gz` sẽ được chuyển sang `messages.3.gz`.
4. Nội dung của `messages.1` sẽ được chuyển sang `messages.2.gz`.
5. Nội dung của `messages` sẽ được chuyển sang `messages.1` và `messages` sẽ trống và sẵn sàng để ghi các mục nhật ký mới.

Theo chỉ thị `logrotate` mà chúng ta sẽ thấy ngay sau đây, hãy lưu ý ba tệp nhật ký cũ hơn đã được nén, trong khi hai tệp nhật ký gần đây nhất thì lại không. Ngoài ra, chúng ta sẽ giữ lại nhật ký từ 4-5 tuần qua. Để đọc các thông báo cách đây 1 tuần, chúng ta sẽ tham khảo `messages.1` (v.v.).

`logrotate` được chạy như một tiến trình tự động hoặc một công việc định kỳ hàng ngày thông qua tệp lệnh `/etc/cron.daily/logrotate` và việc đọc tệp cấu hình `/etc/logrotate.conf`. Tệp này bao gồm một số tùy chọn chung và được chú thích chi tiết với mỗi tùy chọn được giới thiệu bằng một phần giải thích ngắn gọn về mục đích của nó:

```
carol@debian:~$ sudo less /etc/logrotate.conf
# see "man logrotate" for details
# rotate log files weekly
```

```

weekly

# keep 4 weeks worth of backlogs
rotate 4

# create new (empty) log files after rotating old ones
create

# uncomment this if you want your log files compressed
#compress

# packages drop log rotation information into this directory
include /etc/logrotate.d

(...)

```

Như có thể thấy, các tệp cấu hình trong `/etc/logrotate.d` cho các gói cụ thể cũng được bao gồm. Các tệp này chứa — phần lớn — các định nghĩa cục bộ và chỉ định các tệp nhật ký cần được xoay vòng (hãy nhớ rằng các định nghĩa cục bộ sẽ được ưu tiên hơn các định nghĩa toàn cục và các định nghĩa sau sẽ ghi đè lên các định nghĩa trước đó). Sau đây là một đoạn trích của một định nghĩa trong `/etc/logrotate.d/rsyslog`:

```

/var/log/messages
{
    rotate 4
    weekly
    missingok
    notifempty
    compress
    delaycompress
    sharedscripts
    postrotate
        invoke-rc.d rsyslog rotate > /dev/null
    endscrip
}

```

Như có thể thấy, mọi chỉ thị sẽ đều được phân tách khỏi giá trị của nó bằng khoảng trắng và/hoặc một dấu bằng tùy chọn (=). Tuy nhiên, các dòng giữa `postrotate` và `endscript` lại phải tự xuất hiện trên các dòng. Lời giải thích cho việc này như sau:

rotate 4

Giữ lại nhật ký của 4 tuần.

weekly

Xoay vòng các tệp nhật ký mỗi tuần.

missingok

Không đưa ra thông báo lỗi nếu thiếu tệp nhật ký; đi đến tệp nhật ký tiếp theo.

notifempty

Không xoay vòng nhật ký nếu nó trống.

compress

Nén tệp nhật ký bằng `gzip` (mặc định).

delaycompress

Trì hoãn việc nén tệp nhật ký trước đó sang chu kỳ xoay vòng tiếp theo (chỉ hiệu quả khi sử dụng kết hợp với nén). Nó rất hữu ích khi một chương trình không thể bị yêu cầu đóng tệp nhật ký của nó và do đó có thể sẽ tiếp tục ghi vào tệp nhật ký trước đó trong một khoảng thời gian.

sharedscripts

Liên quan đến tệp lệnh *trước xoay vòng* (*prerotate*) và *sau xoay vòng* (*postrotate*). Để ngăn một tệp lệnh được thực thi nhiều lần, hãy chỉ chạy các tệp lệnh một lần bất kể có bao nhiêu tệp nhật ký khớp với một mẫu nhất định (ví dụ như `/var/log/mail/*`). Tuy nhiên, các tệp lệnh sẽ không được chạy nếu không có nhật ký nào trong mẫu cần xoay vòng. Ngoài ra, nếu tệp lệnh thoát ra với lỗi thì mọi hành động còn lại sẽ không được thực thi đối với bất kỳ một nhật ký nào.

postrotate

Cho biết sự bắt đầu của một tệp lệnh *sau xoay vòng*.

invoke-rc.d rsyslog rotate > /dev/null

Sử dụng `/bin/sh` để chạy `invoke-rc.d rsyslog rotate > /dev/null` sau khi xoay vòng nhật ký.

endscript

Chỉ phần cuối của tệp lệnh *sau xoay vòng*.

NOTE

Để có danh sách đầy đủ các chỉ thị và giải thích, hãy tham khảo trang hướng dẫn của `logrotate.conf`.

Bộ đệm vòng Hạt nhân

Vì hạt nhân sẽ tạo ra một số thông báo trước khi `rsyslogd` có sẵn khi khởi động nên một cơ chế để ghi lại các thông báo đó sẽ trở nên cần thiết. Đây là lúc *bộ đệm vòng hạt nhân* phát huy tác dụng. Đó là cấu trúc dữ liệu có kích thước cố định và — do đó — khi nó phát triển cùng với các thông báo mới, các thông báo cũ nhất sẽ biến mất.

Lệnh `dmesg` được sử dụng để in bộ đệm vòng hạt nhân. Do kích thước của bộ đệm nên lệnh này thường được sử dụng kết hợp với tiện ích lọc văn bản `grep`. Ví dụ: để tìm kiếm các thông báo liên quan đến thiết bị USB:

```
root@debian:~# dmesg | grep "usb"
[ 1.241182] usbcore: registered new interface driver usbfs
[ 1.241188] usbcore: registered new interface driver hub
[ 1.250968] usbcore: registered new device driver usb
[ 1.339754] usb usb1: New USB device found, idVendor=1d6b, idProduct=0001, bcdDevice=
4.19
[ 1.339756] usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber=1
(...)
```


Bài tập Hướng dẫn

1. Bạn sẽ sử dụng những tiện ích/lệnh nào trong các tình huống sau:

Mục đích và tệp nhật ký	Tiện ích
Đọc <code>/var/log/syslog.7.gz</code>	
Đọc <code>/var/log/syslog</code>	
Lọc từ <code>renewal</code> trong <code>/var/log/syslog</code>	
Đọc <code>/var/log/faillog</code>	
Đọc động <code>/var/log/syslog</code>	

2. Hãy sắp xếp lại các mục nhập nhật ký sau theo cấu trúc thích hợp để thể hiện một thông báo hợp lệ:

- `debian-server`
- `sshd`
- `[515]:`
- `Sep 13 21:47:56`
- `Server listening on 0.0.0.0 port 22`

Thứ tự đúng sẽ là:

3. Bạn sẽ thêm những quy tắc nào vào `/etc/rsyslog.conf` để thực hiện từng tác vụ sau:

- Gửi tất cả các thư từ tiện ích đại diện `mail` và một mức ưu tiên/mức độ nghiêm trọng là `crit` (trở lên) tới `/var/log/mail.crit`:

- Gửi tất cả các thông báo từ tiện ích đại diện `mail` với mức độ ưu tiên là `alert` và `emergency` tới `/var/log/mail.surgical`:

- Ngoại trừ những thông báo đến từ tiện ích đại diện `cron` và `ntp`, hãy gửi tất cả thông báo—bất kể tiện ích đại diện và mức độ ưu tiên của chúng là gì—tới `/var/log/allmessages`:

- Với tất cả các cài đặt được yêu cầu được cấu hình đúng cách, hãy gửi tất cả các thư từ tiện ích đại diện mail đến máy chủ từ xa có địa chỉ IP là 192.168.1.88 bằng TCP và chỉ định cổng mặc định:

- Không quan trọng tiện ích đại diện của chúng là gì, hãy gửi tất cả các thông báo có mức độ ưu tiên warning (chỉ với mức độ ưu tiên warning) đến /var/log/warnings để ngăn chặn việc ghi quá nhiều vào đĩa:

4. Hãy xem xét đoạn trích sau từ /etc/logrotate.d/samba và giải thích các tùy chọn khác nhau:

```
carol@debian:~$ sudo head -n 11 /etc/logrotate.d/samba
/var/log/samba/log.smbd {
    weekly
    missingok
    rotate 7
    postrotate
        [ ! -f /var/run/samba/smbd.pid ] || /etc/init.d/smbd reload > /dev/null
    endscript
    compress
    delaycompress
    notifempty
}
```

Tùy chọn	Ý nghĩa
weekly	
missingok	
rotate 7	
postrotate	
endscript	
compress	
delaycompress	
notifyempty	

Bài tập Mở rộng

- Trong phần “Khuôn mẫu và Điều kiện lọc”, chúng ta đã sử dụng *bộ lọc dựa trên biểu thức* làm điều kiện lọc. *Bộ lọc dựa trên đặc tính* là một loại bộ lọc khác dành riêng cho `rsyslogd`. Hãy dịch *bộ lọc dựa trên biểu thức* thành một *bộ lọc dựa trên đặc tính*:

Bộ lọc dựa trên biểu thức	Bộ lọc dựa trên đặc tính
<pre>if \$FROMHOST-IP== '192.168.1.4' then ?RemoteLogs</pre>	

- `omusrmsg` là một mô-đun `rsyslog` tích hợp sẵn hỗ trợ việc thông báo cho người dùng (nó sẽ gửi thông báo nhật ký đến cửa sổ dòng lệnh của người dùng). Hãy viết một quy tắc để gửi tất cả các thông báo *khẩn cấp* của tất cả các tiện ích đại diện cho cả `root` và người dùng thông thường `carol`.

Tóm tắt

Trong bài học này, chúng ta đã học về:

- Ghi nhật ký là việc rất quan trọng trong quản trị hệ thống.
- `rsyslogd` là tiện ích chịu trách nhiệm giữ cho các bản ghi được gọn gàng và ngăn nắp.
- Một số dịch vụ sẽ tự xử lý nhật ký của chúng.
- Nói một cách đại khái, nhật ký có thể được phân loại thành nhật ký hệ thống và nhật ký dịch vụ/chương trình.
- Có một số tiện ích thuận tiện cho việc đọc nhật ký: `less`, `more`, `zless`, `zmore`, `grep`, `head` và `tail`.
- Hầu hết các tệp nhật ký đều là các tệp văn bản thuần túy; tuy nhiên, có một số lượng nhỏ tệp nhật ký là tệp nhị phân.
- Liên quan đến nhật ký, `rsyslogd` sẽ nhận thông tin liên quan từ các tệp đặc biệt (ổ nổi và bộ nhớ đệm) trước khi xử lý nó.
- Để phân loại nhật ký, `rsyslogd` sử dụng các quy tắc trong `/etc/rsyslog.conf` hoặc `/etc/rsyslog.d/*`.
- Bất kỳ người dùng nào cũng có thể nhập thông báo của riêng họ vào nhật ký hệ thống theo cách thủ công bằng tiện ích `logger`.
- `rsyslog` cho phép người dùng lưu giữ tất cả các nhật ký trên các mạng IP trong một máy chủ nhật ký tập trung.
- Các khuôn mẫu rất hữu ích cho việc định dạng động tên tệp nhật ký.
- Việc xoay vòng nhật ký có hai mục đích: ngăn chặn các nhật ký cũ sử dụng quá nhiều dung lượng đĩa và giúp người dùng quản lý nhật ký để tham khảo.

Đáp án Bài tập Hướng dẫn

1. Bạn sẽ sử dụng những tiện ích/lệnh nào trong các tình huống sau:

Mục đích và tệp nhật ký	Tiện ích
Đọc <code>/var/log/syslog.7.gz</code>	<code>zmore</code> hoặc <code>zless</code>
Đọc <code>/var/log/syslog</code>	<code>more</code> hoặc <code>less</code>
Lọc từ <code>renewal</code> trong <code>/var/log/syslog</code>	<code>grep</code>
Đọc <code>/var/log/faillog</code>	<code>faillog -a</code>
Đọc động <code>/var/log/syslog</code>	<code>tail -f</code>

2. Hãy sắp xếp lại các mục nhật ký sau theo cách chúng thể hiện thông một thông báo nhật ký hợp lệ với cấu trúc tiêu chuẩn:

- `debian-server`
- `sshd`
- `[515]:`
- `Sep 13 21:47:56`
- `Server listening on 0.0.0.0 port 22`

Thứ tự đúng sẽ là:

```
Sep 13 21:47:56 debian-server sshd[515]: Server listening on 0.0.0.0 port 22
```

3. Bạn sẽ thêm những quy tắc nào vào `/etc/rsyslog.conf` để thực hiện từng tác vụ sau:

- Gửi tất cả các thư từ tiện ích đại diện mail và một mức ưu tiên/mức độ nghiêm trọng là `crit` (trở lên) tới `/var/log/mail.crit`:

```
mail.crit                /var/log/mail.crit
```

- Gửi tất cả các thông báo từ tiện ích đại diện mail với mức độ ưu tiên là `alert` và `emergency` tới `/var/log/mail.surgical`:

```
mail.alert                /var/log/mail.urgent
```

- Ngoài trừ những thông báo đến từ tiện ích đại diện `cron` và `ntp`, hãy gửi tất cả thông báo—bất kể tiện ích đại diện và mức độ ưu tiên của chúng là gì—tới `/var/log/allmessages`:

```
*.*;cron.none;ntp.none /var/log/allmessages
```

- Với tất cả các cài đặt được yêu cầu được cấu hình đúng cách, hãy gửi tất cả các thư từ tiện ích đại diện `mail` đến máy chủ từ xa có địa chỉ IP là `192.168.1.88` bằng TCP và chỉ định cổng mặc định:

```
mail.* @@192.168.1.88:514
```

- Không quan trọng tiện ích đại diện của chúng là gì, hãy gửi tất cả các thông báo có mức độ ưu tiên `warning` (chỉ với mức độ ưu tiên `warning`) đến `/var/log/warnings` để ngăn chặn việc ghi quá nhiều vào đĩa:

```
*.=warning -/var/log/warnings
```

1. Hãy xem xét đoạn trích sau từ `/etc/logrotate.d/samba` và giải thích các tùy chọn khác nhau:

```
carol@debian:~$ sudo head -n 11 /etc/logrotate.d/samba
/var/log/samba/log.smbd {
    weekly
    missingok
    rotate 7
    postrotate
        [ ! -f /var/run/samba/smbd.pid ] || /etc/init.d/smbd reload > /dev/null
    endscript
    compress
    delaycompress
    notifempty
}
```

Tùy chọn	Ý nghĩa
<code>weekly</code>	Xoay vòng các tệp nhật ký mỗi tuần.
<code>missingok</code>	Không đưa ra thông báo lỗi nếu thiếu nhật ký; tiếp tục đi tới tệp tiếp theo.

Tùy chọn	Ý nghĩa
<code>rotate 7</code>	Giữ nhật ký của 7 tuần
<code>postrotate</code>	Chạy tệp lệnh trên dòng sau sau khi xoay vòng nhật ký.
<code>endscript</code>	Cho biết phần cuối của tệp lệnh <i>sau xoay vòng</i> .
<code>compress</code>	Nén nhật ký bằng <code>gzip</code> .
<code>delaycompress</code>	Kết hợp với <code>compress</code> để trì hoãn quá trình nén sang chu kỳ xoay vòng tiếp theo.
<code>notifyempty</code>	Không xoay vòng nhật ký nếu nó trống.

Đáp án Bài tập Mở rộng

- Trong phần “Khuôn mẫu và Điều kiện lọc”, chúng ta đã sử dụng *bộ lọc dựa trên biểu thức* làm điều kiện lọc. *Bộ lọc dựa trên đặc tính* là một loại bộ lọc khác dành riêng cho `rsyslogd`. Hãy dịch *bộ lọc dựa trên biểu thức* thành một *bộ lọc dựa trên đặc tính*:

Bộ lọc dựa trên biểu thức	Bộ lọc dựa trên đặc tính
<code>if \$FROMHOST-IP=='192.168.1.4' then ?RemoteLogs</code>	<code>:fromhost-ip, isequal, "192.168.1.4" ?RemoteLogs</code>

- `omusrmsg` là một mô-đun `rsyslog` tích hợp sẵn hỗ trợ việc thông báo cho người dùng (nó sẽ gửi thông báo nhật ký đến cửa sổ dòng lệnh của người dùng). Hãy viết một quy tắc để gửi tất cả các thông báo *khẩn cấp* của tất cả các tiện ích đại diện cho cả `root` và người dùng thông thường `carol`.

```
*.emerg                                :omusrmsg:root,carol
```




108.2 Bài 2

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	108 Dịch vụ Hệ thống thiết yếu
Mục tiêu:	108.2 Ghi nhật ký Hệ thống
Bài:	2 trên 2

Giới thiệu

Với việc `systemd` được sử dụng bởi hầu hết các bản phân phối chính, sổ nhật ký (`systemd-journal`) đã trở thành một dịch vụ ghi nhật ký tiêu chuẩn. Trong bài học này, chúng ta sẽ thảo luận về cách hoạt động của chương trình này và cách áp dụng nó để thực hiện một số tác vụ: truy vấn, lọc thông tin theo nhiều tiêu chí, định cấu hình bộ nhớ và kích thước của chương trình, xóa dữ liệu cũ, truy xuất dữ liệu của chương trình từ hệ thống cứu hộ hoặc sao chép hệ thống tệp và — cuối cùng nhưng không kém phần quan trọng — hiểu về tương tác của nó với `rsyslogd`.

Khái niệm cơ bản về `systemd`

Được giới thiệu lần đầu tiên trong Fedora, `systemd` đã dần dần thay thế SysV Init để trở thành trình quản lý dịch vụ và hệ thống *tiêu chuẩn* trong hầu hết các bản phân phối Linux chính. Các điểm mạnh của nó là:

- Dễ cấu hình: các tệp đơn vị trái ngược với các tệp lệnh SysV Init.
- Quản lý linh hoạt: ngoài trình nền và các tiến trình, nó còn quản lý các thiết bị, ổ nối và điểm gắn kết.

- Khả năng tương thích ngược với cả SysV Init và Upstart.
- Tải song song trong quá trình khởi động: các dịch vụ sẽ được tải song song trái ngược với việc Sysv Init tải chúng một cách tuần tự.
- Nó có tính năng ghi nhật ký dịch vụ được gọi là *sổ nhật ký* (journal) với những ưu điểm sau:
 - Nó tập trung tất cả các bản ghi ở một nơi.
 - Nó không yêu cầu xoay vòng nhật ký.
 - Nhật ký có thể bị vô hiệu hóa, được tải vào RAM hoặc được lưu giữ bền vững.

Đơn vị và Mục tiêu

systemd hoạt động trên *các đơn vị*. Một đơn vị là bất kỳ một tài nguyên nào mà systemd có thể quản lý (ví dụ như mạng, bluetooth, v.v.). Các đơn vị — lần lượt — sẽ được điều chỉnh bởi *các tệp đơn vị*. Đây là các tệp văn bản thuần túy nằm trong `/lib/systemd/system` và bao gồm các cài đặt cấu hình — dưới dạng các *phần* (sections) và *chỉ thị* (directives) — để quản lý một tài nguyên cụ thể. Có một số loại đơn vị: `service`, `mount`, `automount`, `swap`, `timer`, `device`, `socket`, `path`, `timer`, `snapshot`, `slice`, `scope` và `target`. Do đó, mọi tên tệp đơn vị đều sẽ tuân theo mẫu `<resource_name>.<unit_type>` (ví dụ: `reboot.service`).

Mục tiêu là một loại đơn vị đặc biệt tương đồng với các mức chạy cổ điển của SysV Init. Điều này là do một *đơn vị mục tiêu* sẽ tập hợp nhiều các tài nguyên khác nhau để thể hiện một trạng thái hệ thống cụ thể (ví dụ: `graphical.target` cũng sẽ tương tự như `runlevel 5`, v.v.). Để kiểm tra mục tiêu hiện tại trong hệ thống, hãy sử dụng lệnh `systemctl get-default`:

```
carol@debian:~$ systemctl get-default
graphical.target
```

Mặt khác, mục tiêu và mức chạy thực thi khác nhau ở chỗ mục tiêu sẽ bao gồm cả mức chạy, trong khi mức chạy thì không. Do đó, một mục tiêu có thể đưa ra các mục tiêu khác — điều mà mức chạy không thể làm được.

NOTE

Lý giải về cách hoạt động của các đơn vị `systemd` nằm ngoài phạm vi của bài học này.

Nhật ký hệ thống: `systemd-journald`

`systemd-journald` là một dịch vụ hệ thống đảm nhiệm việc nhận thông tin ghi nhật ký từ nhiều nguồn khác nhau: thông báo từ hạt nhân, thông báo hệ thống đơn giản và có cấu trúc, đầu ra tiêu chuẩn và lỗi tiêu chuẩn của dịch vụ cũng như hồ sơ kiểm tra từ hệ thống kiểm duyệt hạt nhân con

(để xem thêm chi tiết, hãy xem trang hướng dẫn của `systemd-journald`). Nhiệm vụ của nó là tạo ra và duy trì một sổ nhật ký có cấu trúc và các chỉ mục.

Tệp cấu hình của nó là `/etc/systemd/journald.conf` và—như với bất kỳ một dịch vụ nào khác—chúng ta có thể sử dụng lệnh `systemctl` để *khởi động*, *tái khởi động*, *dừng* hoặc—đơn giản nhất—kiểm tra *trạng thái* của nó :

```
root@debian:~# systemctl status systemd-journald
systemd-journald.service - Journal Service
  Loaded: loaded (/lib/systemd/system/systemd-journald.service; static; vendor preset:
enabled)
  Active: active (running) since Sat 2019-10-12 13:43:06 CEST; 5min ago
    Docs: man:systemd-journald.service(8)
          man:journald.conf(5)
 Main PID: 178 (systemd-journal)
  Status: "Processing requests..."
   Tasks: 1 (limit: 4915)
  CGroup: /system.slice/systemd-journald.service
          └─178 /lib/systemd/systemd-journald
(...)
```

Chúng ta cũng có thể sử dụng các tệp cấu hình thuộc loại `journal.conf.d/*.conf`—có thể bao gồm các cấu hình dành riêng cho gói—(hãy tham khảo trang hướng dẫn của `journald.conf` để tìm hiểu thêm).

Nếu được kích hoạt, nhật ký có thể được lưu trữ bền vững trên đĩa hoặc theo một phương thức không ổn định trên hệ thống tệp dựa trên RAM. Nhật ký không phải là một tệp văn bản thuần túy mà một là tệp nhị phân. Vì vậy, chúng ta không thể sử dụng các công cụ phân tích văn bản như `less` hoặc `more` để đọc nội dung của nó; thay vào đó, chúng ta sẽ sử dụng lệnh `journalctl`.

Truy vấn nội dung Sổ Nhật ký

`journalctl` là tiện ích mà chúng ta sẽ sử dụng để truy vấn sổ nhật ký `systemd`. Chúng ta phải là siêu người dùng hoặc sử dụng `sudo` để gọi nó. Nếu được truy vấn mà không có tùy chọn, nó sẽ in toàn bộ sổ nhật ký theo trình tự thời gian (với các mục cũ nhất sẽ được liệt kê đầu tiên):

```
root@debian:~# journalctl
-- Logs begin at Sat 2019-10-12 13:43:06 CEST, end at Sat 2019-10-12 14:19:46 CEST. --
Oct 12 13:43:06 debian kernel: Linux version 4.9.0-9-amd64 (debian-kernel@lists.debian.org)
(...)
Oct 12 13:43:06 debian kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.9.0-9-amd64
root=UUID=b6be6117-5226-4a8a-bade-2db35ccf4cf4 ro qu
```

```
(...)
```

Chúng ta có thể thực hiện các truy vấn cụ thể hơn bằng cách sử dụng một số khoá chuyển:

-r

Các thông báo của sổ nhật ký sẽ được in theo thứ tự ngược:

```
root@debian:~# journalctl -r
-- Logs begin at Sat 2019-10-12 13:43:06 CEST, end at Sat 2019-10-12 14:30:30 CEST. --
Oct 12 14:30:30 debian sudo[1356]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
Oct 12 14:30:30 debian sudo[1356]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -r
Oct 12 14:19:53 debian sudo[1348]: pam_unix(sudo:session): session closed for user root
(...)
```

-f

Nó sẽ in các thông báo nhật ký gần đây nhất và tiếp tục in các mục mới khi chúng được thêm vào nhật ký—giống như `tail -f`:

```
root@debian:~# journalctl -f
-- Logs begin at Sat 2019-10-12 13:43:06 CEST. --
(...)
Oct 12 14:44:42 debian sudo[1356]: pam_unix(sudo:session): session closed for user root
Oct 12 14:44:44 debian sudo[1375]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -f
Oct 12 14:44:44 debian sudo[1375]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
(...)
```

-e

Nó sẽ nhảy đến cuối sổ nhật ký để hiển thị các mục mới nhất trong từng trang đầu ra:

```
root@debian:~# journalctl -e
(...)
Oct 12 14:44:44 debian sudo[1375]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -f
Oct 12 14:44:44 debian sudo[1375]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
```

```
Oct 12 14:45:57 debian sudo[1375]: pam_unix(sudo:session): session closed for user root
Oct 12 14:48:39 debian sudo[1378]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -e
Oct 12 14:48:39 debian sudo[1378]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
```

-n <value>, --lines=<value>

Nó sẽ in số dòng (value) gần đây nhất (nếu không chỉ định <value>, số dòng mặc định sẽ là 10):

```
root@debian:~# journalctl -n 5
(...)
Oct 12 14:44:44 debian sudo[1375]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -f
Oct 12 14:44:44 debian sudo[1375]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
Oct 12 14:45:57 debian sudo[1375]: pam_unix(sudo:session): session closed for user root
Oct 12 14:48:39 debian sudo[1378]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -e
Oct 12 14:48:39 debian sudo[1378]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
```

-k, --dmesg

Tương đương với việc sử dụng lệnh `dmesg`:

```
root@debian:~# journalctl -k
-- Logs begin at Sat 2019-10-12 13:43:06 CEST, end at Sat 2019-10-12 14:53:20 CEST. --
Oct 12 13:43:06 debian kernel: Linux version 4.9.0-9-amd64 (debian-
kernel@lists.debian.org) (gcc version 6.3.0 20170516 (Debian 6.3.0-18
Oct 12 13:43:06 debian kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.9.0-9-amd64
root=UUID=b6be6117-5226-4a8a-bade-2db35ccf4cf4 ro qu
Oct 12 13:43:06 debian kernel: x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating
point registers'
Oct 12 13:43:06 debian kernel: x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
(...)
```

Điều hướng và tìm kiếm trong Sổ Nhật ký

Chúng ta có thể điều hướng đầu ra của sổ nhật ký bằng:

- PageUp, PageDown và các phím mũi tên để di chuyển lên, xuống, trái và phải.

- `>` để đi đến cuối đầu ra.
- `<` để đi đến phần đầu của đầu ra.

Chúng ta có thể tìm kiếm các chuỗi tiến và lùi từ vị trí hiện tại:

- Tìm kiếm tiến: Nhấn `/` và nhập chuỗi cần tìm kiếm, sau đó nhấn Enter.
- Tìm kiếm lùi: Nhấn `?` và nhập chuỗi cần tìm, sau đó nhấn Enter.

Để điều hướng qua các kết quả trùng khớp được tìm thấy, hãy sử dụng `N` để chuyển đến lần xuất hiện tiếp theo của kết quả khớp và `Shift + N` để chuyển đến lần xuất hiện trước đó.

Lọc dữ liệu Sổ Nhật ký

Sổ Nhật ký cho phép chúng ta lọc dữ liệu nhật ký theo các tiêu chí khác nhau:

Số khởi động

`--list-boots`

Nó liệt kê tất cả các phiên khởi động có sẵn. Đầu ra sẽ bao gồm ba cột: cột đầu tiên chỉ định số khởi động (0 là phiên khởi động hiện tại, -1 là phiên trước đó, -2 phiên trước đó nữa, v.v.); cột thứ hai là ID khởi động; cột thứ ba là dấu thời gian:

```
root@debian:~# journalctl --list-boots
0 83df3e8653474ea5aed19b41cdb45b78 Sat 2019-10-12 18:55:41 CEST–Sat 2019-10-12
19:02:24 CEST
```

`-b, --boot`

Nó hiển thị tất cả các thông báo từ phiên khởi động hiện tại. Để xem các thông báo nhật ký từ những phiên khởi động trước, chúng ta chỉ cần thêm tham số phần bù như đã được giải thích ở trên. Ví dụ: để in các thông báo từ phiên khởi động trước, chúng ta sẽ gõ `journalctl -b -1`. Tuy nhiên, hãy nhớ rằng, để khôi phục thông tin từ các nhật ký trước đó, tính bền vững của nhật ký phải được kích hoạt (chúng ta sẽ tìm hiểu cách thực hiện việc này trong phần tiếp theo):

```
root@debian:~# journalctl -b -1
Specifying boot ID has no effect, no persistent journal was found
```

Mức độ ưu tiên

`-p`

Điều thú vị là chúng ta cũng có thể lọc dữ liệu theo mức độ nghiêm trọng/mức độ ưu tiên

bảng tùy chọn `-p`:

```
root@debian:~# journalctl -b -0 -p err
-- No entries --
```

Sổ Nhật ký sẽ thông báo cho chúng ta rằng — cho đến nay — chưa có bất kỳ thông báo nào có mức độ ưu tiên là `error` (hoặc cao hơn) từ phiên khởi động hiện tại. Lưu ý: `-b -0` có thể được bỏ qua khi đề cập đến phiên khởi động hiện tại.

NOTE

Hãy tham khảo bài học trước để biết danh sách đầy đủ của tất cả các mức độ nghiêm trọng của `syslog` (hay còn gọi là mức độ ưu tiên).

Quãng thời gian

Chúng ta có thể yêu cầu `journalctl` chỉ in các thông báo được ghi trong một khung thời gian cụ thể bằng cách sử dụng các khóa chuyển `--since` (từ khi) và `--until` (cho đến khi). Thông số ngày phải tuân theo định dạng `YYYY-MM-DD HH:MM:SS`. Hệ thống sẽ giả định thời điểm là nửa đêm nếu chúng ta bỏ qua yếu tố thời gian. Tương tự, nếu ngày bị bỏ qua, hệ thống sẽ tự giả định là ngày hiện tại. Ví dụ: để xem các thông báo được ghi từ 7:00 tối đến 7:01 tối, chúng ta sẽ gõ:

```
root@debian:~# journalctl --since "19:00:00" --until "19:01:00"
-- Logs begin at Sat 2019-10-12 18:55:41 CEST, end at Sat 2019-10-12 20:10:50 CEST. --
Oct 12 19:00:14 debian systemd[1]: Started Run anacron jobs.
Oct 12 19:00:14 debian anacron[1057]: Anacron 2.3 started on 2019-10-12
Oct 12 19:00:14 debian anacron[1057]: Normal exit (0 jobs run)
Oct 12 19:00:14 debian systemd[1]: anacron.timer: Adding 2min 47.988096s random time.
```

Tương tự như vậy, chúng ta có thể sử dụng thông số thời gian hơi khác một chút: `"integer time-unit ago"` (đơn vị thời gian số nguyên trước đó). Vì vậy, để xem các thông báo được ghi hai phút trước, chúng ta sẽ gõ `sudo journalctl --since "2 minutes ago"`. Chúng ta cũng có thể sử dụng `+` và `-` để chỉ định thời gian liên quan đến thời gian hiện tại (ví dụ: `--since "-2 minutes"` cũng sẽ tương đương với `--since "2 minutes ago"`).

Ngoài các biểu thức số, chúng ta có thể chỉ định một số từ khóa:

yesterday

Tính từ nửa đêm của ngày trước ngày hiện tại.

today

Tính từ nửa đêm của ngày hiện tại.

tomorrow

Tính từ nửa đêm của ngày sau ngày hiện tại.

now

Thời điểm hiện tại.

Hãy cùng xem tất cả các thông báo từ nửa đêm qua cho đến 21:00 tối hôm nay:

```
root@debian:~# journalctl --since "today" --until "21:00:00"
-- Logs begin at Sat 2019-10-12 20:45:29 CEST, end at Sat 2019-10-12 21:06:15 CEST. --
Oct 12 20:45:29 debian sudo[1416]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root
; COMMAND=/bin/systemctl r
Oct 12 20:45:29 debian sudo[1416]: pam_unix(sudo:session): session opened for user
root by carol(uid=0)
Oct 12 20:45:29 debian systemd[1]: Stopped Flush Journal to Persistent Storage.
(...)
```

NOTE

Để tìm hiểu thêm về các cú pháp khác nhau cho thông số thời gian, hãy tham khảo trang hướng dẫn của `systemd.time`.

Chương trình

Để xem các thông báo nhật ký liên quan đến một tệp thực thi cụ thể, cú pháp sau sẽ được sử dụng: `journalctl _/path/to/executable_`:

```
root@debian:~# journalctl /usr/sbin/sshd
-- Logs begin at Sat 2019-10-12 20:45:29 CEST, end at Sat 2019-10-12 21:54:49 CEST. --
Oct 12 21:16:28 debian sshd[1569]: Accepted password for carol from 192.168.1.65 port
34050 ssh2
Oct 12 21:16:28 debian sshd[1569]: pam_unix(sshd:session): session opened for user carol
by (uid=0)
Oct 12 21:16:54 debian sshd[1590]: Accepted password for carol from 192.168.1.65 port
34052 ssh2
Oct 12 21:16:54 debian sshd[1590]: pam_unix(sshd:session): session opened for user carol
by (uid=0)
```

Đơn vị

Hãy nhớ rằng một đơn vị là bất kỳ một tài nguyên nào được `systemd` xử lý và chúng ta cũng có thể lọc theo chúng.

-u

Nó hiển thị thông báo về một đơn vị được chỉ định:

```
root@debian:~# journalctl -u ssh.service
-- Logs begin at Sun 2019-10-13 10:50:59 CEST, end at Sun 2019-10-13 12:22:59 CEST. --
Oct 13 10:51:00 debian systemd[1]: Starting OpenBSD Secure Shell server...
Oct 13 10:51:00 debian sshd[409]: Server listening on 0.0.0.0 port 22.
Oct 13 10:51:00 debian sshd[409]: Server listening on :: port 22.
(...)
```

NOTE

Để in tất cả các đơn vị đã tải và đang hoạt động, hãy sử dụng `systemctl list-units`; để xem tất cả các tệp đơn vị đã được cài đặt, hãy sử dụng `systemctl list-unit-files`.

Trường

Sổ Nhật ký cũng có thể được lọc theo các *trường* (field) cụ thể thông qua bất kỳ cú pháp nào sau đây:

- `<field-name>=<value>`
- `_<field-name>=<value>_`
- `__<field-name>=<value>`

PRIORITY=

Một trong tám giá trị ưu tiên `syslog` có thể được định dạng dưới dạng chuỗi thập phân:

```
root@debian:~# journalctl PRIORITY=3
-- Logs begin at Sun 2019-10-13 10:50:59 CEST, end at Sun 2019-10-13 14:30:50 CEST. --
Oct 13 10:51:00 debian avahi-daemon[314]: chroot.c: open() failed: No such file or directory
```

Hãy lưu ý rằng chúng ta cũng có thể đạt được kết quả tương tự bằng cách sử dụng lệnh `sudo journalctl -p err` đã có ở trên.

SYSLOG_FACILITY=

Bất kỳ mã tiện ích đại diện nào có thể được định dạng dưới dạng chuỗi thập phân. Ví dụ: để xem tất cả các thông báo ở cấp độ người dùng:

```
root@debian:~# journalctl SYSLOG_FACILITY=1
```

```
-- Logs begin at Sun 2019-10-13 10:50:59 CEST, end at Sun 2019-10-13 14:42:52 CEST.
--
Oct 13 10:50:59 debian mtp-probe[227]: checking bus 1, device 2:
"/sys/devices/pci0000:00/0000:00:06.0/usb1/1-1"
Oct 13 10:50:59 debian mtp-probe[227]: bus: 1, device: 2 was not an MTP device
Oct 13 10:50:59 debian mtp-probe[238]: checking bus 1, device 2:
"/sys/devices/pci0000:00/0000:00:06.0/usb1/1-1"
Oct 13 10:50:59 debian mtp-probe[238]: bus: 1, device: 2 was not an MTP device
```

`_PID=`

Hiển thị các thông báo được tạo bởi một ID tiến trình cụ thể. Để xem tất cả các thông báo được tạo bởi `systemd`, hãy gõ:

```
root@debian:~# journalctl _PID=1
-- Logs begin at Sun 2019-10-13 10:50:59 CEST, end at Sun 2019-10-13 14:50:15 CEST.
--
Oct 13 10:50:59 debian systemd[1]: Mounted Debug File System.
Oct 13 10:50:59 debian systemd[1]: Mounted POSIX Message Queue File System.
Oct 13 10:50:59 debian systemd[1]: Mounted Huge Pages File System.
Oct 13 10:50:59 debian systemd[1]: Started Remount Root and Kernel File Systems.
Oct 13 10:50:59 debian systemd[1]: Starting Flush Journal to Persistent Storage...
(...)
```

`_BOOT_ID`

Dựa trên ID khởi động của nó, chúng ta có thể chọn ra các thông báo từ một phiên khởi động cụ thể, ví dụ như `sudo journalctl _BOOT_ID=83df3e8653474ea5aed19b41cdb45b78`.

`_TRANSPORT`

Hiển thị các thông báo nhận được từ một phương tiện cụ thể. Các giá trị có thể là: `audit` (hệ thống con kiểm duyệt hạt nhân), `driver` (được tạo nội bộ), `syslog` (ổ nối syslog), `journal` (giao thức sổ nhật ký gốc), `stdout` (đầu ra tiêu chuẩn của dịch vụ hoặc lỗi tiêu chuẩn), `kernel` (bộ đệm vòng hạt nhân—giống như `dmesg`, `journalctl -k` hoặc `journalctl --dmesg`):

```
root@debian:~# journalctl _TRANSPORT=journal
-- Logs begin at Sun 2019-10-13 20:19:58 CEST, end at Sun 2019-10-13 20:46:36 CEST.
--
Oct 13 20:19:58 debian systemd[1]: Started Create list of required static device nodes for the current kernel.
Oct 13 20:19:58 debian systemd[1]: Starting Create Static Device Nodes in /dev...
```

```
Oct 13 20:19:58 debian systemd[1]: Started Create Static Device Nodes in /dev.
Oct 13 20:19:58 debian systemd[1]: Starting udev Kernel Device Manager...
(...)
```

Kết hợp các Trường

Các trường sẽ không loại trừ lẫn nhau nên chúng ta có thể sử dụng nhiều trường trong cùng một phiên truy vấn. Tuy nhiên, sẽ chỉ những thông báo đồng thời khớp với giá trị của cả hai trường mới được hiển thị:

```
root@debian:~# journalctl PRIORITY=3 SYSLOG_FACILITY=0
-- No entries --
root@debian:~# journalctl PRIORITY=4 SYSLOG_FACILITY=0
-- Logs begin at Sun 2019-10-13 20:19:58 CEST, end at Sun 2019-10-13 20:21:55 CEST. --
Oct 13 20:19:58 debian kernel: acpi PNP0A03:00: fail to add MMCONFIG information, can't
access extended PCI configuration (...)
```

Trừ khi chúng ta sử dụng dấu phân cách + để kết hợp hai biểu thức theo một biểu thức logic *OR*:

```
root@debian:~# journalctl PRIORITY=3 + SYSLOG_FACILITY=0
-- Logs begin at Sun 2019-10-13 20:19:58 CEST, end at Sun 2019-10-13 20:24:02 CEST. --
Oct 13 20:19:58 debian kernel: Linux version 4.9.0-9-amd64 (debian-kernel@lists.debian.org)
(...9
Oct 13 20:19:58 debian kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.9.0-9-amd64
root=UUID= (...)
(...)
```

Mặt khác, chúng ta cũng có thể cung cấp hai giá trị cho cùng một trường và tất cả các mục nhập khớp với một trong hai giá trị sẽ được hiển thị:

```
root@debian:~# journalctl PRIORITY=1
-- Logs begin at Sun 2019-10-13 17:16:24 CEST, end at Sun 2019-10-13 17:30:14 CEST. --
-- No entries --
root@debian:~# journalctl PRIORITY=1 PRIORITY=3
-- Logs begin at Sun 2019-10-13 17:16:24 CEST, end at Sun 2019-10-13 17:32:12 CEST. --
Oct 13 17:16:27 debian connmand[459]: __connman_inet_get_pnp_nameservers: Cannot read /pro
Oct 13 17:16:27 debian connmand[459]: The name net.connman.vpn was not provided by any .se
```

NOTE

Các trường sổ nhật ký có thể thuộc bất kỳ một danh mục nào sau đây: “Trường sổ nhật ký người dùng”, “Trường sổ nhật ký tin cậy”, “Trường nhật ký hạt nhân”,

“Trường thay mặt cho một chương trình khác” và “Trường địa chỉ”. Để biết thêm thông tin về chủ đề này — bao gồm danh sách đầy đủ của các trường — hãy xem trang hướng dẫn của `systemd.journal-fields(7)`.

Mục nhập thủ công vào Sổ Nhật ký Hệ thống: `systemd-cat`

Giống như việc lệnh `logger` được sử dụng để gửi các thông báo từ dòng lệnh đến nhật ký hệ thống (như chúng ta đã thấy trong bài học trước), lệnh `systemd-cat` cũng phục vụ một mục đích tương tự — nhưng hoàn thiện hơn — với sổ nhật ký hệ thống. Nó cho phép chúng ta gửi đầu vào tiêu chuẩn (*stdin*), đầu ra tiêu chuẩn (*stdout*) và lỗi (*stderr*) tới sổ nhật ký.

Nếu được gọi mà không có tham số, lệnh sẽ gửi mọi thứ nó đọc được từ *stdin* tới sổ nhật ký. Khi đã hoàn tất, hãy nhấn `Ctrl + C`:

```
carol@debian:~$ systemd-cat
This line goes into the journal.
^C
```

Nếu nó được chuyển qua đầu ra của lệnh theo đường dẫn, lệnh này cũng sẽ được gửi đến sổ nhật ký:

```
carol@debian:~$ echo "And so does this line." | systemd-cat
```

Nếu theo sau là một lệnh, đầu ra của lệnh đó cũng sẽ được gửi đến nhật ký — cùng với *stderr* (nếu có):

```
carol@debian:~$ systemd-cat echo "And so does this line too."
```

Ngoài ra, chúng ta cũng có thể chỉ định mức độ ưu tiên với tùy chọn `-p`:

```
carol@debian:~$ systemd-cat -p emerg echo "This is not a real emergency."
```

Hãy tham khảo trang hướng dẫn của `systemd-cat` để tìm hiểu về các tùy chọn khác của nó.

Để xem bốn dòng cuối cùng trong sổ nhật ký:

```
carol@debian:~$ journalctl -n 4
(...)
-- Logs begin at Sun 2019-10-20 13:43:54 CEST. --
```

```
Nov 13 23:14:39 debian cat[1997]: This line goes into the journal.
Nov 13 23:19:16 debian cat[2027]: And so does this line.
Nov 13 23:23:21 debian echo[2030]: And so does this line too.
Nov 13 23:26:48 debian echo[2034]: This is not a real emergency.
```

NOTE

Các mục số nhật ký có mức độ ưu tiên *khẩn cấp* (emergency) sẽ được in màu đỏ đậm trên hầu hết các hệ thống.

Lưu trữ Nhật ký bền vững

Như đã đề cập ở trên, chúng ta có ba tùy chọn khi nói đến vị trí của sổ nhật ký:

- Có thể vô hiệu hoá hoàn toàn tính năng ghi sổ nhật ký (tuy nhiên, ta vẫn có thể chuyển hướng đến các tiện ích khác như bảng điều khiển).
- Giữ sổ nhật ký trong bộ nhớ — điều này sẽ làm cho nó linh hoạt hơn — và loại bỏ nhật ký với mỗi phiên tái khởi động hệ thống. Trong trường hợp này, thư mục `/run/log/journal` sẽ được tạo và sử dụng.
- Làm cho sổ nhật ký trở nên bền vững hơn để nó có thể ghi nhật ký vào đĩa. Trong trường hợp này, thông báo nhật ký sẽ đi vào thư mục `/var/log/journal`.

Hành vi mặc định sẽ như sau: nếu `/var/log/journal/` không tồn tại, nhật ký sẽ được lưu theo một cách "tạm bợ" vào một thư mục trong `/run/log/journal/` và — do đó — sẽ bị mất khi hệ thống khởi động lại. Tên của thư mục — `/etc/machine-id` — là một chuỗi chữ thường thập lục phân, kết thúc bởi một dòng mới và có 32 ký tự:

```
carol@debian:~$ ls /run/log/journal/8821e1fdf176445697223244d1dfbd73/
system.journal
```

Nếu thử đọc bằng lệnh `less`, chúng ta sẽ nhận được một cảnh báo; thay vào đó, hãy sử dụng lệnh `journalctl`:

```
root@debian:~# less /run/log/journal/9a32ba45ce44423a97d6397918de1fa5/system.journal
"/run/log/journal/9a32ba45ce44423a97d6397918de1fa5/system.journal" may be a binary file.
See it anyway?
root@debian:~# journalctl
-- Logs begin at Sat 2019-10-05 21:26:38 CEST, end at Sat 2019-10-05 21:31:27 CEST. --
(...)
Oct 05 21:26:44 debian systemd-journald[1712]: Runtime journal
(/run/log/journal/9a32ba45ce44423a97d6397918de1fa5) is 4.9M, max 39.5M, 34.6M free.
Oct 05 21:26:44 debian systemd[1]: Started Journal Service.
```

(...)

Nếu `/var/log/journal/` tồn tại, nhật ký sẽ được lưu trữ bền vững ở đó. Nếu thư mục này bị xóa, `systemd-journald` sẽ không tạo lại nó mà thay vào đó sẽ ghi vào `/run/log/journal`. Ngay sau khi chúng ta tạo lại `/var/log/journal/` và khởi động lại trình nền, việc ghi nhật ký bền vững sẽ được thiết lập lại:

```
root@debian:~# mkdir /var/log/journal/
root@debian:~# systemctl restart systemd-journald
root@debian:~# journalctl
(...)
Oct 05 21:33:49 debian systemd-journald[1712]: Received SIGTERM from PID 1 (systemd).
Oct 05 21:33:49 debian systemd[1]: Stopped Journal Service.
Oct 05 21:33:49 debian systemd[1]: Starting Journal Service...
Oct 05 21:33:49 debian systemd-journald[1768]: Journal started
Oct 05 21:33:49 debian systemd-journald[1768]: System journal
(/var/log/journal/9a32ba45ce44423a97d6397918de1fa5) is 8.0M, max 1.1G, 1.1G free.
Oct 05 21:33:49 debian systemd[1]: Started Journal Service.
Oct 05 21:33:49 debian systemd[1]: Starting Flush Journal to Persistent Storage...
(...)
```

NOTE

Theo mặc định, hệ thống sẽ có các tệp nhật ký cụ thể cho mỗi người dùng đã đăng nhập nằm trong `/var/log/journal/`. Do đó—cùng với các tệp `system.journal`—chúng ta cũng sẽ tìm thấy các tệp thuộc loại `user-1000.journal`.

Ngoài những gì vừa được đề cập, cách trình nền nhật ký xử lý việc lưu trữ nhật ký có thể được thay đổi sau khi cài đặt bằng cách điều chỉnh tệp cấu hình của nó là `/etc/systemd/journald.conf`. Tùy chọn chính là `Storage=` và nó có thể có các giá trị sau:

Storage=volatile

Dữ liệu nhật ký sẽ được lưu trữ riêng trong bộ nhớ—trong `/run/log/journal/`. Nếu không có trong hệ thống, thư mục sẽ được tạo tự động.

Storage=persistent

Theo mặc định, dữ liệu nhật ký sẽ được lưu trữ trên đĩa—trong `/var/log/journal/`—với một bản dự phòng vào bộ nhớ (`/run/log/journal/`) trong giai đoạn tiền khởi động nếu đĩa đó không thể ghi được. Cả hai thư mục sẽ được tạo tự động nếu cần.

Storage=auto

`auto` cũng tương tự như `persistent` nhưng thư mục `/var/log/journal` sẽ không được tạo tự

động nếu cần. Đây là hành vi mặc định.

Storage=none

Tất cả dữ liệu nhật ký sẽ bị loại bỏ. Tuy nhiên, việc chuyển tiếp đến các mục tiêu khác như bảng điều khiển, bộ đệm nhật ký hạt nhân hoặc ổ nối syslog vẫn có thể thực hiện được.

Ví dụ: để `systemd-journald` tạo `/var/log/journal/` và chuyển sang bộ lưu trữ bền vững, chúng ta sẽ chỉnh sửa `/etc/systemd/journald.conf` và thiết lập `Storage=persistent`, lưu tệp và khởi động lại trình nền bằng `sudo systemctl restart systemd-journald`. Để đảm bảo quá trình khởi động lại diễn ra suôn sẻ, chúng ta luôn có thể kiểm tra trạng thái của trình nền:

```
root@debian:~# systemctl status systemd-journald
systemd-journald.service - Journal Service
  Loaded: loaded (/lib/systemd/system/systemd-journald.service; static; vendor preset:
enabled)
  Active: active (running) since Wed 2019-10-09 10:03:40 CEST; 2s ago
    Docs: man:systemd-journald.service(8)
          man:journald.conf(5)
 Main PID: 1872 (systemd-journal)
  Status: "Processing requests..."
   Tasks: 1 (limit: 3558)
  Memory: 1.1M
  CGroup: /system.slice/systemd-journald.service
          └─1872 /lib/systemd/systemd-journald

Oct 09 10:03:40 debian10 systemd-journald[1872]: Journal started
Oct 09 10:03:40 debian10 systemd-journald[1872]: System journal
(/var/log/journal/9a32ba45ce44423a97d6397918de1fa5) is 8.0M, max 1.2G, 1.2G free.
```

NOTE

Các tệp nhật ký trong `/var/log/journal/<machine-id>/` hoặc `/run/log/journal/<machine-id>/` sẽ có hậu tố `.journal` (ví dụ như `system.journal`). Tuy nhiên, nếu chúng được phát hiện là bị hỏng hoặc trình nền bị dừng theo cách không chính thống, chúng sẽ được đổi tên khi gắn thêm `~` (ví dụ như `system.journal~`) và trình nền sẽ bắt đầu ghi vào một tệp trống mới.

Xóa dữ liệu Sổ Nhật ký cũ: Kích cỡ của Sổ Nhật ký

Nhật ký sẽ được lưu trong các tệp sổ nhật ký có tên tệp được kết thúc bằng `.journal` hoặc `.journal~` và được đặt trong thư mục thích hợp (`/run/log/journal` hoặc `/var/log/journal` như được định cấu hình). Để kiểm tra xem các tệp nhật ký đang chiếm bao nhiêu dung lượng đĩa (cả được lưu trữ và đang hoạt động), hãy sử dụng khóa chuyển `--disk-usage`:

```
root@debian:~# journalctl --disk-usage
Archived and active journals take up 24.0M in the filesystem.
```

Nhật ký `systemd` mặc định có kích thước tối đa là 10% kích thước của hệ thống tệp nơi chúng được lưu trữ. Chẳng hạn, trên hệ thống tệp 1GB, chúng sẽ không chiếm quá 100 MB. Khi đạt đến giới hạn này, các nhật ký cũ sẽ bắt đầu biến mất để duy trì mức gần giá trị này.

Tuy nhiên, việc tuân thủ giới hạn kích thước trên các tệp sổ nhật ký được lưu trữ có thể được quản lý bằng cách điều chỉnh một loạt các tùy chọn cấu hình trong `/etc/systemd/journald.conf`. Các tùy chọn này được chia thành hai loại tùy thuộc vào loại hệ thống tệp được sử dụng: bền vững - persistent (`/var/log/journal`) hoặc trong bộ nhớ - in-memory (`/run/log/journal`). Tùy chọn bền vững sử dụng các tùy chọn có tiền tố `System` và sẽ chỉ được áp dụng nếu tính năng ghi nhật ký bền vững được kích hoạt đúng cách và sau khi hệ thống được khởi động hoàn toàn. Đối với tùy chọn trong bộ nhớ, tên tùy chọn sẽ bắt đầu bằng `Runtime` và sẽ áp dụng trong các trường hợp sau:

SystemMaxUse=, RuntimeMaxUse=

Chúng kiểm soát dung lượng đĩa mà sổ nhật ký có thể chiếm dụng. Nó mặc định là 10% kích thước hệ thống tệp nhưng có thể được sửa đổi (ví dụ: `SystemMaxUse=500M`), miễn là không vượt quá mức tối đa là 4GiB.

SystemKeepFree=, RuntimeKeepFree=

Chúng kiểm soát dung lượng đĩa trống dành cho những người dùng khác. Nó mặc định là 15% kích thước hệ thống tệp nhưng có thể được sửa đổi (ví dụ: `SystemKeepFree=500M`), miễn là không vượt quá mức tối đa là 4GiB.

Về quyền ưu tiên của `*MaxUse` và `*KeepFree`, `systemd-journald` sẽ đáp ứng cả hai bằng cách sử dụng giá trị nhỏ hơn trong hai giá trị. Tương tự như vậy, hãy nhớ rằng chỉ các tệp nhật ký được lưu trữ (trái ngược với đang hoạt động) mới bị xóa.

SystemMaxFileSize=, RuntimeMaxFileSize=

Chúng kiểm soát kích thước tối đa mà các tệp sổ nhật ký riêng lẻ có thể đạt tới. Giá trị mặc định sẽ là 1/8 của `*MaxUse`. Việc giảm kích thước được thực hiện một cách đồng bộ và các giá trị có thể được chỉ định theo byte hoặc sử dụng K, M, G, T, P, E tương ứng cho Kibibytes, Mebibytes, Gibibyte, Tebibytes, Pebibytes và Exbibytes.

SystemMaxFiles=, RuntimeMaxFiles=

Chúng thiết lập số lượng tối đa các tệp sổ nhật ký riêng lẻ và được lưu trữ để lưu trữ (các tệp sổ nhật ký đang hoạt động sẽ không bị ảnh hưởng). Giá trị mặc định sẽ là 100.

Ngoài việc xóa và xoay vòng các thông báo nhật ký dựa trên kích thước, `systemd-journald` còn

cho phép các tiêu chí dựa trên thời gian bằng cách sử dụng hai tùy chọn sau: `MaxRetentionSec=` và `MaxFileSec=`. Hãy tham khảo trang hướng dẫn của `journald.conf` để biết thêm thông tin về các tùy chọn này và các tùy chọn khác.

NOTE

Bất cứ khi nào bạn sửa đổi hành vi mặc định của `systemd-journald` bằng cách bỏ chú thích và chỉnh sửa các tùy chọn trong `/etc/systemd/journald.conf`, bạn sẽ đều phải khởi động lại trình nền để các thay đổi có hiệu lực.

Dọn dẹp Sổ Nhật ký

Chúng ta có thể xóa thủ công các tệp sổ nhật ký đã được lưu trữ tại bất kỳ một thời điểm nào bằng bất kỳ tùy chọn nào trong ba tùy chọn sau:

--vacuum-time=

Tùy chọn dựa trên thời gian này sẽ loại bỏ tất cả các thông báo trong các tệp sổ nhật ký có dấu thời gian cũ hơn khung thời gian đã chỉ định. Các giá trị phải được viết bằng bất kỳ một hậu tố nào sau đây: `s`, `m`, `h`, `days` (hoặc `d`), `months`, `weeks` (hoặc `w`) và `year` (hoặc `y`). Ví dụ: để loại bỏ tất cả các thông báo trong các tệp sổ nhật ký đã lưu trữ lâu hơn 1 tháng:

```
root@debian:~# journalctl --vacuum-time=1months
Deleted archived journal
/var/log/journal/7203088f20394d9c8b252b64a0171e08/system@27dd08376f71405a91794e632ede97ed-0000000000000001-00059475764d46d6.journal (16.0M).
Deleted archived journal /var/log/journal/7203088f20394d9c8b252b64a0171e08/user-1000@e7020d80d3af42f0bc31592b39647e9c-0000000000000008e-00059479df9677c8.journal (8.0M).
```

--vacuum-size=

Tùy chọn dựa trên kích thước này sẽ xóa các tệp sổ nhật ký đã lưu trữ cho đến khi chúng chiếm một giá trị dưới mức kích thước đã chỉ định (`size`). Các giá trị phải được ghi với một trong các hậu tố sau đây: `K`, `M`, `G` hoặc `T`. Ví dụ: để loại bỏ các tệp sổ nhật ký được lưu trữ cho đến khi chúng đạt tới dưới 100 Mebibyte:

```
root@debian:~# journalctl --vacuum-size=100M
Vacuuming done, freed 0B of archived journals from
/run/log/journal/9a32ba45ce44423a97d6397918de1fa5.
```

--vacuum-files=

Tùy chọn này sẽ đảm bảo số tệp nhật ký được giữ lại không vượt quá số lượng đã chỉ định. Giá trị của nó sẽ là một số nguyên. Ví dụ: để giới hạn số lượng tệp sổ nhật ký được lưu trữ ở mức 10:

```
root@debian:~# journalctl --vacuum-files=10
Vacuuming done, freed 0B of archived journals from
/run/log/journal/9a32ba45ce44423a97d6397918de1fa5.
```

Việc dọn dẹp sẽ chỉ loại bỏ các tệp sổ nhật ký đã được lưu trữ. Nếu muốn loại bỏ mọi thứ (bao gồm cả các tệp sổ nhật ký đang hoạt động), chúng ta cần sử dụng một tín hiệu (SIGUSR2) để yêu cầu xoay vòng ngay lập tức các tệp sổ nhật ký bằng tùy chọn `--rotate`. Các tín hiệu quan trọng khác có thể được gọi bằng các tùy chọn sau:

`--flush (SIGUSR1)`

Nó sẽ yêu cầu xóa các tệp sổ nhật ký từ `/run/` sang `/var/` để làm cho nhật ký được bền vững, kích hoạt tính năng ghi nhật ký bền vững và `/var/` phải được gắn kết.

`--sync (SIGRTMIN+1)`

Tùy chọn này được sử dụng để yêu cầu ghi tất cả các dữ liệu nhật ký chưa được ghi vào đĩa.

NOTE

Để kiểm tra tính nhất quán cục bộ của tệp sổ nhật ký hãy sử dụng `journalctl` với tùy chọn `--verify`. Chúng ta sẽ thấy một thanh tiến trình khi quá trình kiểm tra hoàn tất và mọi vấn đề có thể xảy ra sẽ được hiển thị.

Truy xuất dữ liệu Sổ Nhật ký từ Hệ thống Cứu hộ

Với tư cách là quản trị viên hệ thống, chúng ta có thể bắt gặp tình huống cần phải truy cập các tệp sổ nhật ký trên ổ cứng của một máy bị lỗi thông qua hệ thống cứu hộ (một khóa CD hoặc USB có thể khởi động có chứa một bản phân phối Linux trực tiếp).

`journalctl` sẽ tìm kiếm các tệp sổ nhật ký trong `/var/log/journal/<machine-id>/`. Vì ID máy trên hệ thống cứu hộ và hệ thống bị lỗi sẽ khác nhau nên chúng ta phải sử dụng tùy chọn sau:

`-D </path/to/dir>, --directory=</path/to/dir>`

Với tùy chọn này, chúng ta sẽ chỉ định một đường dẫn thư mục mà tại đó, `journalctl` sẽ tìm kiếm các tệp sổ nhật ký thay vì vị trí hệ thống và thời gian chạy mặc định.

Vì vậy, điều cần thiết là phải gắn `rootfs (/dev/sda1)` của hệ thống bị lỗi vào hệ thống tệp của hệ thống cứu hộ và tiến hành đọc các tệp sổ nhật ký như sau:

```
root@debian:~# journalctl -D /media/carol/faulty.system/var/log/journal/
-- Logs begin at Sun 2019-10-20 12:30:45 CEST, end at Sun 2019-10-20 12:32:57 CEST. --
oct 20 12:30:45 suse-server kernel: Linux version 4.12.14-lp151.28.16-default
(geeko@buildhost) (...)
oct 20 12:30:45 suse-server kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.12.14-
```

```
lp151.28.16-default root=UUID=7570f67f-4a08-448e-aa09-168769cb9289 splash=>
oct 20 12:30:45 suse-server kernel: x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating
point registers'
oct 20 12:30:45 suse-server kernel: x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
(...)
```

Các tùy chọn khác có thể sẽ hữu ích trong trường hợp này là:

-m, --merge

Chúng hợp nhất các mục từ tất cả các sổ nhật ký có sẵn trong `/var/log/journal`, bao gồm cả các sổ nhật ký từ xa.

--file

Nó sẽ hiển thị các mục trong một tệp cụ thể, ví dụ như `journalctl --file /var/log/journal/64319965bda04dfa81d3bc4e7919814a/user-1000.journal`.

--root

Một đường dẫn thư mục có nghĩa là thư mục gốc được truyền dưới dạng đối số. `journalctl` sẽ tìm kiếm các tệp sổ nhật ký ở đó (ví dụ: `journalctl --root /faulty.system/`).

Hãy xem trang hướng dẫn của `journalctl` để biết thêm thông tin.

Chuyển tiếp dữ liệu Nhật ký tới một Trình nền syslog truyền thống

Dữ liệu nhật ký từ sổ nhật ký có thể được chuyển tiếp tới một trình nền `syslog` truyền thống bằng cách:

- Chuyển tiếp các thông báo đến tệp ổ nối `/run/systemd/journal/syslog` để `syslogd` đọc. Tiện ích này được kích hoạt với tùy chọn `ForwardToSyslog=yes`.
- Có một trình nền `syslog` hoạt động giống như `journalctl`, từ đó đọc thông báo nhật ký trực tiếp từ các tệp sổ nhật ký. Trong trường hợp này, tùy chọn liên quan là `Storage` và nó phải có giá trị khác `none`.

NOTE

Tương tự, chúng ta cũng có thể chuyển tiếp các thông báo nhật ký đến các đích đến khác bằng các tùy chọn sau: `ForwardToKMsg` (bộ đệm nhật ký hạt nhân — `kmsg`), `ForwardToConsole` (bảng điều khiển hệ thống) hoặc `ForwardToWall` (tất cả người dùng đã đăng nhập qua `wall`). Để biết thêm thông tin, hãy tham khảo trang hướng dẫn về `journald.conf`.

Bài tập Hướng dẫn

1. Giả sử bạn là `root`, hãy hoàn thành bảng sau với lệnh `journalctl` thích hợp:

Mục đích	Lệnh
In các mục nhập hạt nhân	
In các thông báo từ lần khởi động thứ hai bắt đầu từ phần đầu của sổ nhật ký	
In thông báo từ lần khởi động thứ hai bắt đầu từ phần cuối của sổ nhật ký	
In các thông báo nhật ký gần đây nhất và tiếp tục theo dõi những thông báo mới	
Kể từ bây giờ chỉ in các thông báo mới và cập nhật đầu ra liên tục	
In các thông báo từ lần khởi động trước với mức độ ưu tiên là <code>warning</code> và theo thứ tự ngược	

2. Hành vi của sổ nhật ký liên quan đến việc lưu trữ hầu hết được kiểm soát bởi giá trị của tùy chọn `Storage` trong `/etc/systemd/journald.conf`. Hãy cho biết hành vi nào có liên quan đến giá trị nào trong bảng sau:

Hành vi	<code>Storage=auto</code>	<code>Storage=none</code>	<code>Storage=persistent</code>	<code>Storage=volatile</code>
Dữ liệu nhật ký bị loại bỏ nhưng có thể chuyển tiếp.				
Khi hệ thống đã khởi động xong, dữ liệu nhật ký sẽ được lưu trữ trong <code>/var/log/journal</code> . Nếu chưa có thì thư mục sẽ được tạo.				

Hành vi	Storage=auto	Storage=none	Storage=persistent	Storage=volatile
Khi hệ thống đã khởi động xong, dữ liệu nhật ký sẽ được lưu trữ trong <code>/var/log/journal</code> . Nếu chưa có thì thư mục sẽ không được tạo.				
Dữ liệu nhật ký sẽ được lưu trữ trong <code>/var/run/journal</code> nhưng sẽ không tồn tại khi khởi động lại.				

3. Như bạn đã biết, sổ nhật ký có thể được dọn dẹp theo cách thủ công dựa trên thời gian, kích thước và số lượng tệp. Hãy hoàn thành các tác vụ sau bằng cách sử dụng `journalctl` và các tùy chọn thích hợp:

- Kiểm tra xem các tệp sổ nhật ký đã chiếm bao nhiêu dung lượng đĩa:

- Giảm số lượng không gian dành riêng cho các tệp sổ nhật ký được lưu trữ và đặt thành 200MiB:

- Kiểm tra lại dung lượng đĩa và giải thích kết quả:

Bài tập Mở rộng

1. Bạn nên sửa đổi những tùy chọn nào trong `/etc/systemd/journald.conf` để các thông báo được chuyển tiếp đến `/dev/tty5`? Các tùy chọn nên có những giá trị gì?

2. Hãy cho biết bộ lọc `journalctl` chính xác để in các thông tin sau:

Mục đích	Bộ lọc + Giá trị
In thông báo của một người dùng cụ thể	
In thông báo từ máy chủ có tên <code>debian</code>	
In thông báo thuộc một nhóm cụ thể	
In thông báo thuộc về <code>root</code>	
Dựa trên đường dẫn thực thi, in thông báo <code>sudo</code>	
Dựa vào tên lệnh, in thông báo <code>sudo</code>	

3. Khi lọc theo mức độ ưu tiên, các nhật ký có mức độ ưu tiên cao hơn mức được chỉ định cũng sẽ được đưa vào danh sách; ví dụ: `journalctl -p err` sẽ in cả các thông báo ở mức *error*, *critical*, *alert* và *emergency*. Tuy nhiên, bạn có thể yêu cầu `journalctl` chỉ hiển thị một phạm vi cụ thể. Bạn sẽ sử dụng lệnh nào để yêu cầu `journalctl` chỉ in các thông báo ở mức độ ưu tiên *warning*, *error* và *critical*?

4. Mức độ ưu tiên cũng có thể được xác định bằng số. Hãy viết lại lệnh trong bài tập trước bằng cách sử dụng biểu diễn dạng số của mức độ ưu tiên:

Tóm tắt

Trong bài học này, chúng ta đã học về:

- Ưu điểm của việc sử dụng `systemd` làm trình quản lý hệ thống và dịch vụ.
- Khái niệm cơ bản về các đơn vị và mục tiêu `systemd`.
- Nơi `systemd-journald` lấy dữ liệu ghi nhật ký.
- Các tùy chọn có thể truyền cho `systemctl` để điều khiển `systemd-journald`: `start`, `status`, `restart` và `stop`.
- Nơi chứa tệp cấu hình của sổ nhật ký — `/etc/systemd/journald.conf` — và các tùy chọn chính của nó.
- Cách truy vấn sổ nhật ký một cách tổng quát và tìm dữ liệu cụ thể bằng cách sử dụng các bộ lọc.
- Cách điều hướng và tìm kiếm trong sổ nhật ký.
- Cách xử lý việc lưu trữ tệp sổ nhật ký: trong bộ nhớ và trên đĩa.
- Làm thế nào để vô hiệu hóa hoàn toàn việc ghi sổ nhật ký.
- Cách kiểm tra dung lượng đĩa mà sổ nhật ký đã sử dụng, thực thi giới hạn kích thước đối với các tệp sổ nhật ký được lưu trữ và dọn dẹp các tệp sổ nhật ký đã lưu trữ theo cách thủ công (*vacuuming*).
- Cách lấy dữ liệu sổ nhật ký từ một hệ thống cứu hộ.
- Cách chuyển tiếp dữ liệu nhật ký sang một trình nền `syslog` truyền thống.

Các lệnh được sử dụng trong bài học này:

`systemctl`

Kiểm soát trình quản lý dịch vụ và hệ thống `systemd`.

`journalctl`

Truy vấn sổ nhật ký `systemd`.

`1`

Liệt kê nội dung thư mục.

`less`

Xem nội dung tệp

mkdir

Tạo thư mục.

Đáp án Bài tập Hướng dẫn

1. Giả sử bạn là `root`, hãy hoàn thành bảng sau với lệnh `journalctl` thích hợp:

Mục đích	Lệnh
In các mục nhập hạt nhân	<code>journalctl -k</code> hoặc <code>journalctl --dmesg</code>
In các thông báo từ lần khởi động thứ hai bắt đầu từ phần đầu của sổ nhật ký	<code>journalctl -b 2</code>
In thông báo từ lần khởi động thứ hai bắt đầu từ phần cuối của sổ nhật ký	<code>journalctl -b -2 -r</code> hoặc <code>journalctl -r -b -2</code>
In các thông báo nhật ký gần đây nhất và tiếp tục theo dõi những thông báo mới	<code>journalctl -f</code>
Kể từ bây giờ chỉ in các thông báo mới và cập nhật đầu ra liên tục	<code>journalctl --since "now" -f</code>
In các thông báo từ lần khởi động trước với mức độ ưu tiên là <code>warning</code> và theo thứ tự ngược	<code>journalctl -b -1 -p warning -r</code>

2. Hành vi của sổ nhật ký liên quan đến việc lưu trữ hầu hết được kiểm soát bởi giá trị của tùy chọn `Storage` trong `/etc/systemd/journald.conf`. Hãy cho biết hành vi nào có liên quan đến giá trị nào trong bảng sau:

Hành vi	<code>Storage=auto</code>	<code>Storage=none</code>	<code>Storage=persistent</code>	<code>Storage=volatile</code>
Dữ liệu nhật ký bị loại bỏ nhưng có thể chuyển tiếp.		x		
Khi hệ thống đã khởi động xong, dữ liệu nhật ký sẽ được lưu trữ trong <code>/var/log/journal</code> . Nếu chưa có thì thư mục sẽ được tạo.			x	

Hành vi	Storage=auto	Storage=none	Storage=persistent	Storage=volatile
Khi hệ thống đã khởi động xong, dữ liệu nhật ký sẽ được lưu trữ trong <code>/var/log/journal</code> . Nếu chưa có thì thư mục sẽ không được tạo.	x			
Dữ liệu nhật ký sẽ được lưu trữ trong <code>/var/run/journal</code> nhưng sẽ không tồn tại khi khởi động lại.				x

3. Như bạn đã biết, sổ nhật ký có thể được dọn dẹp theo cách thủ công dựa trên thời gian, kích thước và số lượng tệp. Hãy hoàn thành các tác vụ sau bằng cách sử dụng `journalctl` và các tùy chọn thích hợp:

- Kiểm tra xem các tệp sổ nhật ký đã chiếm bao nhiêu dung lượng đĩa:

```
journalctl --disk-usage
```

- Giảm số lượng không gian dành riêng cho các tệp sổ nhật ký được lưu trữ và đặt thành 200MiB:

```
journalctl --vacuum-size=200M
```

- Kiểm tra lại dung lượng đĩa và giải thích kết quả:

```
journalctl --disk-usage
```

Không có mối tương quan ở đây vì `--disk-usage` hiển thị dung lượng bị chiếm bởi cả tệp sổ nhật ký đang hoạt động và được lưu trữ trong khi `--vacuum-size` chỉ áp dụng cho các tệp

được lưu trữ.

Đáp án Bài tập Mở rộng

1. Bạn nên sửa đổi những tùy chọn nào trong `/etc/systemd/journald.conf` để các thông báo được chuyển tiếp đến `/dev/tty5`? Các tùy chọn nên có những giá trị gì?

```
ForwardToConsole=yes
TTYPath=/dev/tty5
```

2. Hãy cho biết bộ lọc `journalctl` chính xác để in các thông tin sau:

Mục đích	Bộ lọc + Giá trị
In thông báo của một người dùng cụ thể	<code>_ID=<user-id></code>
In thông báo từ máy chủ có tên <code>debian</code>	<code>_HOSTNAME=debian</code>
In thông báo thuộc một nhóm cụ thể	<code>_GID=<group-id></code>
In thông báo thuộc về <code>root</code>	<code>_UID=0</code>
Dựa trên đường dẫn thực thi, in thông báo <code>sudo</code>	<code>_EXE=/usr/bin/sudo</code>
Dựa vào tên lệnh, in thông báo <code>sudo</code>	<code>_COMM=sudo</code>

3. Khi lọc theo mức độ ưu tiên, các nhật ký có mức độ ưu tiên cao hơn mức được chỉ định cũng sẽ được đưa vào danh sách; ví dụ: `journalctl -p err` sẽ in cả các thông báo ở mức *error*, *critical*, *alert* và *emergency*. Tuy nhiên, bạn có thể yêu cầu `journalctl` chỉ hiển thị một phạm vi cụ thể. Bạn sẽ sử dụng lệnh nào để yêu cầu `journalctl` chỉ in các thông báo ở mức độ ưu tiên *warning*, *error* và *critical*?

```
journalctl -p warning..crit
```

4. Mức độ ưu tiên cũng có thể được xác định bằng số. Hãy viết lại lệnh trong bài tập trước bằng cách sử dụng biểu diễn dạng số của mức độ ưu tiên:

```
journalctl -p 4..2
```



108.3 Các khái niệm cơ bản về Tác nhân Vận chuyển Thư (MTA)

Tham khảo các mục tiêu LPI

[LPIC-1 version 5.0, Exam 102, Objective 108.3](#)

Khối lượng

3

Các lĩnh vực kiến thức chính

- Tạo bí danh email.
- Cấu hình chuyển tiếp e-mail.
- Kiến thức về các chương trình MTA phổ biến (postfix, sendmail, exim) (không cần cấu hình)

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- `~/ .forward`
- sendmail emulation layer commands
- `newaliases`
- `mail`
- `mailq`
- `postfix`
- `sendmail`
- `exim`



108.1 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	108 Dịch vụ Hệ thống thiết yếu
Mục tiêu:	108.3 Các khái niệm cơ bản về Tác nhân vận chuyển Thư (MTA)
Bài:	1 trên 1

Giới thiệu

Trong các hệ điều hành giống Unix (chẳng hạn như Linux), mỗi một người dùng đều có *hộp thư đến* của riêng mình: đây là một vị trí đặc biệt trên hệ thống tệp mà những người dùng khác (ngoại trừ siêu người dùng) không thể truy cập được; nó được sử dụng để lưu trữ các thư điện tử cá nhân của người dùng. Thư mới sẽ được thêm vào hộp thư đến của người dùng bởi *Tác nhân vận chuyển Thư* (MTA). MTA là một chương trình chạy dưới dạng một dịch vụ hệ thống với nhiệm vụ thu thập các thư được gửi tới bởi các tài khoản cục bộ khác cũng như các thư nhận được từ mạng được gửi từ các tài khoản người dùng từ xa.

Chính MTA này cũng chịu trách nhiệm gửi thư tới mạng nếu địa chỉ đích là một tài khoản từ xa. Nó thực hiện điều này bằng cách sử dụng vị trí hệ thống tệp làm *hộp thư đi* cho tất cả mọi người dùng hệ thống: ngay khi người dùng đặt một thư mới vào hộp thư đi, MTA sẽ xác định nút mạng mục tiêu từ tên miền do địa chỉ email đích cung cấp — phần sau dấu @ — và sau đó nó sẽ cố gắng chuyển thư đến MTA từ xa bằng cách sử dụng *Giao thức truyền thư đơn giản* (SMTP). SMTP được thiết kế dành cho các mạng không đáng tin cậy; do đó, nó sẽ cố gắng thiết lập các tuyến phân phối thay thế nếu nút đích đến của thư chính không thể truy cập được.

MTA cục bộ và từ xa

Tài khoản người dùng truyền thống trong các máy được kết nối mạng sẽ tạo nên một kịch bản trao đổi email đơn giản nhất mà trong đó, mỗi nút mạng sẽ chạy trình nền MTA của riêng nó. Chúng ta sẽ không cần có phần mềm nào khác ngoài MTA để gửi và nhận thư. Tuy nhiên, trên thực tế, việc sử dụng tài khoản một tài khoản thư điện tử từ xa và không có dịch vụ MTA cục bộ đang hoạt động lại phổ biến hơn (tức là thay vào đó sẽ sử dụng một ứng dụng email khách để truy cập tài khoản từ xa).

Không giống như các tài khoản cục bộ, một tài khoản email từ xa — còn được gọi là *hộp thư từ xa* — sẽ yêu cầu xác thực người dùng để cấp quyền truy cập vào hộp thư và MTA từ xa (trong trường hợp này được gọi đơn giản là máy chủ *SMTP*). Mặc dù người dùng tương tác với hộp thư đến cục bộ và MTA đã được hệ thống xác định nhưng hệ thống từ xa vẫn sẽ phải xác minh danh tính của người dùng trước khi xử lý thư của nó thông qua IMAP hoặc POP3.

NOTE

Ngày nay, phương pháp phổ biến nhất để gửi và nhận email là thông qua tài khoản được lưu trữ trên máy chủ từ xa (ví dụ như máy chủ thư điện tử tập trung của công ty sẽ lưu trữ tất cả các tài khoản nhân viên hoặc dịch vụ thư điện tử cá nhân như *Gmail* của Google). Thay vì thu thập các thư được gửi cục bộ, ứng dụng email khách sẽ kết nối với hộp thư từ xa và truy xuất các thư từ đó. Các giao thức POP3 và IMAP thường được sử dụng để truy xuất thư từ máy chủ từ xa, nhưng các giao thức độc quyền không chuẩn khác cũng có thể được sử dụng.

Khi một trình nền MTA đang chạy trên hệ thống cục bộ, người dùng cục bộ có thể gửi email đến những người dùng cục bộ khác hoặc tới những người dùng trên máy từ xa, miễn là hệ thống của họ cũng có dịch vụ MTA chấp nhận kết nối mạng. Cổng TCP 25 là cổng tiêu chuẩn cho giao tiếp SMTP, nhưng các cổng khác cũng có thể được sử dụng, tùy thuộc vào lược đồ xác thực và/hoặc mã hóa đang được sử dụng (nếu có).

Bỏ qua các cấu trúc liên kết liên quan đến quyền truy cập vào hộp thư từ xa, mạng trao đổi email giữa các tài khoản người dùng Linux thông thường cũng có thể được triển khai, miễn là tất cả các nút mạng đều có một MTA đang hoạt động và có thể thực hiện các tác vụ sau:

- Duy trì hàng đợi hộp thư đi của các thư sẽ được gửi. Đối với mỗi thư được xếp hàng đợi, MTA cục bộ sẽ đánh giá MTA đích từ địa chỉ của người nhận.
- Giao tiếp với trình nền MTA từ xa bằng cách sử dụng SMTP. MTA cục bộ sẽ có thể sử dụng Giao thức truyền thư đơn giản (SMTP) qua chồng TCP/IP để nhận, gửi và chuyển hướng thư từ/đến các trình nền MTA từ xa khác.
- Duy trì hộp thư đến riêng cho từng tài khoản cục bộ. MTA thường sẽ lưu trữ thư ở định dạng *mbx*: một tệp văn bản duy nhất chứa tất cả các email theo trình tự.

Thông thường, địa chỉ email sẽ chỉ định một tên miền làm vị trí (ví dụ như `lpi.org` trong `info@lpi.org`). Trong trường hợp này, MTA của người gửi sẽ truy vấn dịch vụ DNS để tìm bản ghi MX tương ứng. Bản ghi DNS MX sẽ có chứa địa chỉ IP của MTA xử lý email cho miền đó. Nếu cùng một tên miền có nhiều hơn một bản ghi MX được chỉ định trong DNS, MTA sẽ cố gắng liên hệ với chúng theo các giá trị ưu tiên. Nếu địa chỉ người nhận không chỉ định tên miền hoặc miền không có bản ghi MX thì phần sau ký hiệu `@` sẽ được coi là máy chủ của MTA đích.

Các khía cạnh bảo mật cũng phải được tính đến nếu máy chủ MTA được hiển thị với các máy chủ trên Internet. Ví dụ: một người dùng không xác định có thể sử dụng MTA cục bộ để mạo danh người dùng khác và gửi đi các email có khả năng gây hại. Hiện tượng MTA chuyển tiếp email một cách "mù quáng" được gọi là *chuyển tiếp mở* (open relay), tức là khi nó có thể được sử dụng làm trung gian để có khả năng nguy hại người thực sự đã gửi thư. Để ngăn chặn những hành vi lạm dụng này, người dùng được khuyến nghị là chỉ chấp nhận kết nối từ các miền được ủy quyền và triển khai lược đồ xác thực an toàn.

Ngoài ra, chúng ta còn có nhiều triển khai MTA khác nhau cho Linux, mỗi cách triển khai sẽ tập trung vào các khía cạnh cụ thể như khả năng tương thích, hiệu suất, bảo mật, v.v. Tuy nhiên, tất cả các MTA đều sẽ tuân theo các nguyên tắc cơ bản và cung cấp các tính năng tương tự như nhau.

MTA của Linux

MTA truyền thống có sẵn cho các hệ thống Linux là *Sendmail* - một MTA có mục đích chung rất linh hoạt được nhiều hệ điều hành giống Unix sử dụng. Một số MTA phổ biến khác là *Postfix*, *qmail* và *Exim*. Lý do chính để chọn một MTA thay thế là để triển khai các tính năng nâng cao dễ dàng hơn vì việc định cấu hình máy chủ email tùy chỉnh trong Sendmail có thể là một nhiệm vụ khá phức tạp. Ngoài ra, mỗi bản phân phối cũng có thể có MTA tương thích của riêng nó với các cài đặt được xác định trước phù hợp với hầu hết các thiết lập phổ biến. Tất cả các MTA đều nhắm đến mục đích thay thế Sendmail; vì vậy, tất cả các ứng dụng tương thích với Sendmail đều sẽ hoạt động bất kể là MTA nào đang được sử dụng.

Nếu MTA đang chạy nhưng không chấp nhận kết nối mạng, nó sẽ chỉ có thể gửi email trên máy cục bộ. Đối với MTA *sendmail*, tệp `/etc/mail/sendmail.mc` phải được sửa đổi để chấp nhận các kết nối không cục bộ. Để làm được điều đó, mục nhập

```
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

nên được sửa đổi thành địa chỉ mạng chính xác và dịch vụ sẽ được khởi động lại. Một số bản phân phối Linux (chẳng hạn như Debian) có thể có cung cấp các công cụ cấu hình để giúp hiển thị máy chủ email với một tập hợp các tính năng thường được sử dụng được xác định sẵn.

TIP

Do vấn đề bảo mật, hầu hết các bản phân phối Linux sẽ không cài đặt MTA theo mặc định. Để kiểm tra các ví dụ được đưa ra trong bài học này, hãy đảm bảo rằng mọi máy đều có một MTA đang chạy và chúng chấp nhận kết nối trên cổng TCP 25. Vì mục đích bảo mật, các hệ thống này sẽ không được tiếp xúc với các kết nối đến từ Internet công cộng trong quá trình thử nghiệm.

Khi MTA đang chạy và chấp nhận các kết nối từ mạng, các email mới sẽ được chuyển đến nó bằng các lệnh SMTP được gửi qua kết nối TCP. Lệnh `nc` — một tiện ích mạng được sử dụng để đọc và ghi dữ liệu chung trên mạng — có thể được sử dụng để gửi lệnh SMTP trực tiếp đến MTA. Nếu lệnh `nc` không có, nó sẽ được cài đặt với gói `ncat` hoặc `nmap-ncat`, tùy thuộc vào hệ thống quản lý gói đang được sử dụng. Việc viết các lệnh SMTP trực tiếp vào MTA sẽ giúp chúng ta hiểu rõ hơn về giao thức và các khái niệm email chung khác; ngoài ra, nó cũng có thể giúp chẩn đoán các vấn đề phát sinh trong quá trình gửi thư.

Ví dụ: nếu người dùng `emma` tại máy chủ `lab1.campus` muốn gửi thư cho người dùng `dave` tại máy chủ `lab2.campus` thì cô ấy có thể sử dụng lệnh `nc` để kết nối trực tiếp với `lab2.campus` MTA (giả định là nó đang lắng nghe trên cổng TCP 25):

```
$ nc lab2.campus 25
220 lab2.campus ESMTP Sendmail 8.15.2/8.15.2; Sat, 16 Nov 2019 00:16:07 GMT
HELO lab1.campus
250 lab2.campus Hello lab1.campus [10.0.3.134], pleased to meet you
MAIL FROM: emma@lab1.campus
250 2.1.0 emma@lab1.campus... Sender ok
RCPT TO: dave@lab2.campus
250 2.1.5 dave@lab2.campus... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
Subject: Recipient MTA Test

Hi Dave, this is a test for your MTA.
.
250 2.0.0 xAG0G7Y0000595 Message accepted for delivery
QUIT
221 2.0.0 lab2.campus closing connection
```

Sau khi kết nối được thiết lập, MTA từ xa sẽ tự nhận dạng và sau đó sẽ sẵn sàng nhận lệnh SMTP. Lệnh SMTP đầu tiên trong ví dụ là `HELO lab1.campus` biểu thị `lab1.campus` là người khởi tạo trao đổi. Hai lệnh tiếp theo là `MAIL FROM: emma@lab1.campus` và `RCPT TO: dave@lab2.campus` cho biết người gửi và người nhận. Một email tiêu chuẩn sẽ bắt đầu sau lệnh `DATA` và kết thúc bằng một dấu chấm trên một dòng. Để thêm trường `subject` (chủ đề) vào email,

trường này phải ở dòng đầu tiên sau lệnh DATA như trong ví dụ. Khi sử dụng trường chủ đề, chúng ta phải có một dòng trống ngăn cách với nội dung email. Lệnh QUIT sẽ kết thúc kết nối với MTA tại máy chủ lab2.campus.

Trên máy chủ lab2.campus, người dùng dave sẽ nhận được một thông báo tương tự như You have new mail in /var/spool/mail/dave ngay khi người dùng này vào một phiên vỏ. Tập này sẽ chứa thông báo email thô được gửi bởi emma cũng như các tiêu đề được MTA thêm vào:

```
$ cat /var/spool/mail/dave
From emma@lab1.campus Sat Nov 16 00:19:13 2019
Return-Path: <emma@lab1.campus>
Received: from lab1.campus (lab1.campus [10.0.3.134])
    by lab2.campus (8.15.2/8.15.2) with SMTP id xAG0G7Y0000595
    for dave@lab2.campus; Sat, 16 Nov 2019 00:17:06 GMT
Date: Sat, 16 Nov 2019 00:16:07 GMT
From: emma@lab1.campus
Message-Id: <201911160017.xAG0G7Y0000595@lab2.campus>
Subject: Recipient MTA Test

Hi Dave, this is a test for your MTA.
```

Tiêu đề Received: đã cho thấy rằng thông báo từ lab1.campus đã được lab2.campus nhận trực tiếp. Theo mặc định, MTA sẽ chỉ chấp nhận thư gửi đến người nhận cục bộ. Lỗi sau có thể xảy ra nếu người dùng emma cố gắng gửi email cho người dùng henry tại máy chủ lab3.campus nhưng lại sử dụng MTA lab2.campus thay vì MTA lab3.campus chuẩn:

```
$ nc lab2.campus 25
220 lab2.campus ESMTP Sendmail 8.15.2/8.15.2; Sat, 16 Nov 2019 00:31:44 GMT
HELO lab1.campus
250 lab2.campus Hello lab1.campus [10.0.3.134], pleased to meet you
MAIL FROM: emma@lab1.campus
250 2.1.0 emma@lab1.campus... Sender ok
RCPT TO: henry@lab3.campus
550 5.7.1 henry@lab3.campus... Relaying denied
```

Các số phản hồi SMTP được bắt đầu bằng 5, chẳng hạn như thông báo Relaying denied (Chuyển tiếp bị từ chối) cho biết đã có lỗi. Có những tình huống thực sự cần thiết phải chuyển tiếp, chẳng hạn như khi máy chủ gửi và máy chủ nhận email không được kết nối toàn thời gian: MTA trung gian có thể được định cấu hình để chấp nhận các email dành cho các máy chủ khác và hoạt động như một máy chủ chuyển tiếp SMTP có khả năng chuyển tiếp thư giữa các MTA.

Khả năng định tuyến lưu lượng email thông qua các máy chủ SMTP trung gian không khuyến khích việc kết nối trực tiếp với máy chủ được chỉ định bởi địa chỉ email của người nhận (như minh họa trong các ví dụ trước). Hơn nữa, địa chỉ email thường đều có tên miền để xác nhận vị trí (sau @); vì vậy, tên thực của máy chủ MTA tương ứng phải được truy xuất thông qua DNS. Do đó, người dùng được khuyến nghị nên ủy quyền nhiệm vụ xác định máy chủ đích thích hợp cho MTA cục bộ hoặc cho máy chủ SMTP từ xa khi sử dụng hộp thư từ xa.

Sendmail có cung cấp lệnh `sendmail` để thực hiện nhiều thao tác liên quan đến email bao gồm cả việc hỗ trợ soạn thư mới. Nó cũng yêu cầu người dùng nhập tiêu đề email bằng tay nhưng theo một cách thân thiện hơn so với việc sử dụng trực tiếp các lệnh SMTP. Vì vậy, một phương pháp phù hợp hơn để người dùng `emma@lab1.campus` gửi email đến `dave@lab2.campus` sẽ là:

```
$ sendmail dave@lab2.campus
From: emma@lab1.campus
To: dave@lab2.campus
Subject: Sender MTA Test

Hi Dave, this is a test for my MTA.
.
```

Ở đây lại một lần nữa, dấu chấm trên một dòng đã tự nó kết thúc thư. Thư phải được gửi ngay đến người nhận trừ khi MTA cục bộ không thể liên lạc với MTA từ xa. Lệnh `mailq` (nếu được thực thi bởi siêu người dùng) sẽ hiển thị tất cả các thư chưa được gửi. Ví dụ: nếu MTA tại `lab2.campus` không phản hồi thì lệnh `mailq` sẽ liệt kê thư chưa được gửi và nguyên nhân gây ra lỗi:

```
# mailq
      /var/spool/mqueue (1 request)
-----Q-ID----- --Size--  -----Q-Time-----  -----Sender/Recipient-----
xAIK3D9S000453      36 Mon Nov 18 20:03 <emma@lab1.campus>
                    (Deferred: Connection refused by lab2.campus.)
                    <dave@lab2.campus>
Total requests: 1
```

Vị trí mặc định cho hàng đợi hộp thư đi là `/var/spool/mqueue/`, nhưng các MTA khác nhau có thể sử dụng các vị trí khác nhau trong thư mục `/var/spool/` (ví dụ như việc Postfix sẽ tạo một cây thư mục trong `/var/spool/postfix/` để quản lý hàng đợi). Lệnh `mailq` cũng tương đương với `sendmail -bp` và chúng phải xuất hiện bất kể là MTA nào được cài đặt trong hệ thống. Để đảm bảo khả năng tương thích ngược, hầu hết mọi MTA đều có cung cấp các lệnh quản trị thư truyền thống này.

Nếu máy chủ đích đến email chính — khi nó được cung cấp từ bản ghi MX DNS cho miền — không thể truy cập được, MTA sẽ cố gắng liên hệ với các mục có mức độ ưu tiên thấp hơn (nếu có bất kỳ mục nào được chỉ định). Nếu không mục nào có thể truy cập được, thư sẽ ở lại trong hàng đợi hộp thư đi cục bộ để được gửi đi sau. Nếu được định cấu hình để làm như vậy, MTA có thể kiểm tra định kỳ tính khả dụng của các máy chủ từ xa và thử thực hiện một phiên phân phối mới. Nếu sử dụng MTA tương thích với Sendmail, một phiên mới sẽ ngay lập tức diễn ra bằng lệnh `sendmail -q`.

Sendmail sẽ lưu trữ các thư đến trong một tệp được đặt tên theo chủ sở hữu hộp thư đến tương ứng (ví dụ như `/var/spool/mail/dave`). Các MTA khác (như Postfix) có thể lưu trữ các thư đến ở các vị trí như `/var/mail/dave`, nhưng nội dung tệp sẽ giống như nhau. Trong ví dụ này, lệnh `sendmail` đã được sử dụng trên máy chủ của người gửi để soạn thư; do đó, tiêu đề thư thô sẽ cho thấy rằng email đã thực hiện các bước bổ sung trước khi đến đích đến cuối cùng:

```
$ cat /var/spool/mail/dave
From emma@lab1.campus  Mon Nov 18 20:07:39 2019
Return-Path: <emma@lab1.campus>
Received: from lab1.campus (lab1.campus [10.0.3.134])
    by lab2.campus (8.15.2/8.15.2) with ESMTPS id xAIK7c1C000432
    (version=TLSv1.3 cipher=TLS_AES_256_GCM_SHA384 bits=256 verify=NOT)
    for <dave@lab2.campus>; Mon, 18 Nov 2019 20:07:38 GMT
Received: from lab1.campus (localhost [127.0.0.1])
    by lab1.campus (8.15.2/8.15.2) with ESMTPS id xAIK3D9S000453
    (version=TLSv1.3 cipher=TLS_AES_256_GCM_SHA384 bits=256 verify=NOT)
    for <dave@lab2.campus>; Mon, 18 Nov 2019 20:03:13 GMT
Received: (from emma@localhost)
    by lab1.campus (8.15.2/8.15.2/Submit) id xAIK0doL000449
    for dave@lab2.campus; Mon, 18 Nov 2019 20:00:39 GMT
Date: Mon, 18 Nov 2019 20:00:39 GMT
Message-Id: <201911182000.xAIK0doL000449@lab1.campus>
From: emma@lab1.campus
To: dave@lab2.campus
Subject: Sender MTA Test

Hi Dave, this is a test for my MTA.
```

Từ dưới lên trên, các dòng bắt đầu bằng `Received:` cho biết lộ trình của thư. Thư đã được gửi bởi người dùng emma với lệnh `sendmail dave@lab2.campus` được cấp trên lab1.campus như đã nêu trong tiêu đề `Received:` đầu tiên. Sau đó, vẫn trên lab1.campus, MTA đã sử dụng ESMTPS — một tập lớn của SMTP để bổ sung phần mở rộng mã hóa — để gửi thư đến MTA tại lab2.campus như đã nêu ở tiêu đề `Received:` cuối cùng (trên cùng).

MTA sẽ hoàn thành công việc của mình sau khi thư được lưu trong hộp thư đến của người dùng. Người dùng thường sẽ thực hiện một số loại lọc email, chẳng hạn như trình chặn thư rác hoặc thực thi các quy tắc lọc do người dùng xác định. Các tác vụ này được thực thi bởi các ứng dụng của bên thứ ba hoạt động cùng với MTA. Ví dụ: MTA có thể gọi tiện ích *SpamAssassin* để đánh dấu các thư đáng ngờ bằng các tính năng phân tích văn bản của nó.

Mặc dù có thể thực hiện được nhưng việc đọc trực tiếp tệp hộp thư sẽ không được thuận tiện. Thay vào đó, người dùng nên sử dụng một chương trình email khách (ví dụ như Thunderbird, Evolution hoặc KMail) để phân tích tệp và quản lý thư một cách thích hợp. Các chương trình như vậy cũng cung cấp các tính năng bổ sung như lối tắt cho các hành động phổ biến, thư mục con cho hộp thư đến, v.v.

Lệnh mail và Trình duyệt Thư (MUA)

Chúng ta có thể viết thư trực tiếp ở định dạng thô, nhưng sẽ thực tế hơn nhiều khi sử dụng một ứng dụng khách — hay còn được gọi là MUA (*Mail User Agent* - Tác nhân Người dùng Thư) — để tăng tốc quá trình và tránh sai sót. MUA sẽ đảm nhiệm các công việc không tên, nghĩa là ứng dụng email khách sẽ trình bày và sắp xếp các thư nhận được cũng như xử lý giao tiếp thích hợp với MTA sau khi người dùng soạn email.

Có nhiều loại Tác nhân Người dùng Thư khác nhau. Các ứng dụng dành cho máy tính như *Mozilla Thunderbird* và *Evolution* của Gnome có hỗ trợ cả tài khoản email cục bộ lẫn từ xa. Ngay cả giao diện *Webmail* cũng có thể được coi là một loại MUA vì chúng làm trung gian cho sự tương tác giữa người dùng và MTA cơ bản. Tuy nhiên, ứng dụng email khách sẽ không bị hạn chế ở giao diện đồ họa: từng ứng dụng email khách bằng điều khiển được sử dụng rộng rãi để truy cập các hộp thư không được tích hợp với giao diện đồ họa và để tự động hóa các tác vụ liên quan đến email trong tệp lệnh vỏ

Ban đầu, lệnh `mail` của Unix chỉ nhằm mục đích chia sẻ thư từ giữa những người dùng hệ thống cục bộ (lệnh `mail` đầu tiên có từ phiên bản Unix đầu tiên phát hành năm 1971). Khi trao đổi email trên mạng trở nên phổ biến hơn, các chương trình khác đã được tạo ra để đối phó với các hệ thống phân phối thư mới và dần thay thế chương trình `mail` cũ.

Ngày nay, lệnh `mail` được sử dụng phổ biến nhất được cung cấp bởi gói `mailx` tương thích với tất cả các tính năng email hiện đại. Trong hầu hết các bản phân phối Linux, lệnh `mail` chỉ là một liên kết tượng trưng đến lệnh `mailx`. Các triển khai khác (như gói *GNU Mailutils*) về cơ bản cũng cung cấp các tính năng tương tự như `mailx` với một số khác biệt nhỏ, đặc biệt là về các tùy chọn dòng lệnh.

Bất kể cách triển khai là gì, tất cả các biến thể hiện đại của lệnh `mail` đều hoạt động ở hai chế độ: *chế độ bình thường* và *chế độ gửi*. Nếu một địa chỉ email được cung cấp làm đối số cho lệnh `mail`,

nó sẽ vào chế độ gửi; nếu không, nó sẽ vào chế độ bình thường (chế độ đọc). Ở chế độ bình thường, các thư đã nhận sẽ được liệt kê với một chỉ mục bằng số cho từng thư. Vì vậy, người dùng có thể tham chiếu tới từng chỉ mục khi gõ lệnh trong dấu nhắc lệnh tương tác. Ví dụ: lệnh `print 1` có thể được sử dụng để hiển thị nội dung của thư số 1. Các lệnh tương tác có thể được viết tắt; do đó, các lệnh như `print`, `delete` hoặc `reply` có thể được thay thế tương ứng bằng `p`, `d` hoặc `r`. Lệnh `mail` sẽ luôn xem xét các thư được nhận lần cuối hoặc các thư được xem lần cuối khi số chỉ mục của chúng bị bỏ qua. Lệnh `quit` hoặc `q` sẽ thoát khỏi chương trình.

Chế độ *gửi* sẽ đặc biệt hữu ích trong việc gửi email tự động. Ví dụ: nó có thể được sử dụng để gửi email đến quản trị viên hệ thống nếu tệp lệnh bảo trì theo lịch trình không thực hiện được nhiệm vụ của nó. Trong chế độ gửi, `mail` sẽ sử dụng nội dung từ *đầu vào tiêu chuẩn* làm nội dung thư:

```
$ mail -s "Maintenance fail" henry@lab3.campus <<<"The maintenance script failed at `date`"
```

Trong ví dụ này, tùy chọn `-s` đã được thêm vào để đưa trường chủ đề vào thư. Nội dung của thư được cung cấp bởi chuyển hướng *Hereline* tới đầu vào tiêu chuẩn, nhưng nội dung của tệp hoặc đầu ra của lệnh cũng có thể được dẫn tới *stdin* của chương trình. Nếu chuyển hướng đến đầu vào tiêu chuẩn không cung cấp nội dung thì chương trình sẽ đợi người dùng nhập nội dung thư. Trong trường hợp này, hãy nhấn phím `Ctrl` + `D` để kết thúc thư. Lệnh `mail` sẽ thoát ngay lập tức sau khi thư được thêm vào hàng đợi hộp thư đi.

Tùy chỉnh phát Thư

Theo mặc định, tài khoản email trên hệ thống Linux sẽ được liên kết với tài khoản hệ thống tiêu chuẩn. Ví dụ: Nếu người dùng Carol có tên đăng nhập là `carol` trên máy chủ `lab2.campus` thì địa chỉ email của cô ấy sẽ là `carol@lab2.campus`. Sự liên kết một-một giữa các tài khoản hệ thống và hộp thư có thể được mở rộng bằng các phương pháp tiêu chuẩn được cung cấp bởi hầu hết các bản phân phối Linux, đặc biệt là cơ chế định tuyến email do tệp `/etc/aliases` cung cấp.

Bí danh (alias) email là một người nhận email “ảo” có thư nhận được được chuyển hướng đến hộp thư cục bộ hiện có hoặc đến các loại đích đến lưu trữ hoặc xử lý thư khác. Ví dụ: bí danh rất hữu ích trong việc đặt các thư được gửi tới `postmaster@lab2.campus` (một hộp thư cục bộ thông thường trong hệ thống `lab2.campus`) trong hộp thư của Carol. Để làm như vậy, dòng `postmaster: carol` phải được thêm vào tệp `/etc/aliases` trong `lab2.campus`. Sau khi sửa đổi tệp `/etc/aliases`, lệnh `newaliases` sẽ được thực thi để cập nhật cơ sở dữ liệu bí danh của MTA và làm cho các thay đổi có hiệu lực. Các lệnh `sendmail -bi` hoặc `sendmail -I` cũng có thể được sử dụng để cập nhật cơ sở dữ liệu bí danh.

Mỗi bí danh sẽ được xác định trên mỗi dòng theo định dạng `<alias>: <destination>`. Ngoài các hộp thư cục bộ thông thường được biểu thị bằng tên người dùng tương ứng, chúng ta còn có

các loại đích đến khác:

- Một đường dẫn đầy đủ (bắt đầu bằng /) tới một tệp. Thư được gửi tới bí danh tương ứng sẽ được thêm vào tệp.
- Một lệnh xử lý thư. <destination> phải được bắt đầu bằng một ký tự ống dẫn và nếu lệnh có chứa các ký tự đặc biệt (như khoảng trắng) thì nó phải được đặt trong dấu trích dẫn kép. Ví dụ: bí danh subscribe: |subscribe.sh trong lab2.campus sẽ chuyển tiếp tất cả các thư được gửi đến subscribe@lab2.campus tới đầu vào tiêu chuẩn của lệnh subscribe.sh. Nếu Sendmail đang chạy ở chế độ *vô hạn chế*, các lệnh được phép — hoặc liên kết tới chúng — phải ở dưới dạng /etc/smrsh/.
- Một tệp được bao gồm. Một bí danh có thể có nhiều đích đến (được phân tách bằng dấu phẩy); do đó, việc giữ chúng trong một tệp ngoại vi có thể sẽ thực tế hơn. Từ khóa :include: phải chỉ ra đường dẫn tệp (như trong :include:/var/local/destinations).
- Một địa chỉ ngoại vi. Bí danh cũng có thể chuyển tiếp thư đến các địa chỉ email bên ngoài.
- Một bí danh khác.

Một người dùng cục bộ không có đặc quyền có thể xác định bí danh cho email của riêng họ bằng cách chỉnh sửa tệp .forward trong thư mục chính của họ. Vì bí danh chỉ có thể ảnh hưởng đến hộp thư của chính người dùng đó nên chỉ có phần <destination> là cần thiết. Ví dụ: để chuyển tiếp tất cả các email đến đến một địa chỉ bên ngoài, người dùng dave trong lab2.campus có thể tạo tệp ~/ .forward sau:

```
$ cat ~/ .forward
emma@lab1.campus
```

Nó sẽ chuyển tiếp tất cả các email được gửi đến dave@lab2.campus tới emma@lab1.campus. Giống như tệp /etc/aliases, các quy tắc chuyển hướng khác có thể được thêm vào .forward, mỗi quy tắc trên một dòng. Tuy nhiên, tệp .forward chỉ có thể được ghi bởi chủ sở hữu của nó và ta sẽ không cần phải thực thi lệnh newaliases sau khi sửa đổi nó. Các tệp bắt đầu bằng dấu chấm sẽ không xuất hiện trong danh sách tệp thông thường. Điều này có thể khiến người dùng không nắm rõ được về các bí danh đang hoạt động. Do đó, quan trọng nhất là ta phải xác minh xem tệp có tồn tại hay không khi chẩn đoán sự cố gửi email.

Bài tập Hướng dẫn

1. Nếu không có thêm tùy chọn hoặc đối số, lệnh `mail henry@lab3.campus` sẽ chuyển sang chế độ nhập để người dùng có thể nhập thư gửi tới `henry@lab3.campus`. Sau khi soạn xong thư, phím nào sẽ đóng chế độ nhập và gửi email đi?

2. Siêu người dùng có thể thực thi lệnh nào để liệt kê các thư chưa được gửi bắt nguồn từ hệ thống cục bộ?

3. Làm cách nào một người dùng không có đặc quyền có thể sử dụng phương thức MTA tiêu chuẩn để tự động chuyển tiếp tất cả thư đến của họ tới địa chỉ `dave@lab2.campus`?

Bài tập Mở rộng

1. Bằng cách sử dụng lệnh `mail` do `mailx` cung cấp, lệnh nào sẽ gửi thư đến `emma@lab1.campus` với tệp `logs.tar.gz` dưới dạng tệp đính kèm và đầu ra của lệnh `uname -a` làm nội dung email?

2. Một quản trị viên dịch vụ email muốn giám sát việc lưu chuyển email qua mạng nhưng họ không muốn làm hộp thư của mình lộn xộn với các thư thử nghiệm. Làm cách nào để quản trị viên này có thể định cấu hình bí danh email trên toàn hệ thống để chuyển hướng tất cả email được gửi tới người dùng `test` sang tệp `/dev/null`?

3. Ngoài `newaliases`, lệnh nào có thể được sử dụng để cập nhật cơ sở dữ liệu bí danh sau khi thêm bí danh mới vào `/etc/aliases`?

Tóm tắt

Bài học này đề cập đến vai trò và cách sử dụng Đơn vị vận chuyển Thư (MTA) trong hệ thống Linux. MTA cung cấp một phương thức tiêu chuẩn để liên lạc giữa các tài khoản người dùng và có thể được kết hợp với các phần mềm khác để cung cấp thêm chức năng. Bài học đã thảo luận về các chủ đề sau:

- Các khái niệm về công nghệ, hộp thư và giao thức liên quan đến email.
- Cách các MTA của Linux trao đổi thư qua mạng.
- Bảng điều khiển ứng dụng email khách và MUA (Trình duyệt Thư).
- Bí danh và chuyển tiếp email cục bộ.

Các công nghệ, lệnh và quy trình đã được đề cập tới là:

- SMTP và các giao thức liên quan.
- MTA có sẵn cho Linux: Sendmail, Postfix, qmail, Exim.
- Lệnh MTA và MUA: `sendmail` và `mail`.
- Các tệp và lệnh quản trị: `mailq`, `/etc/aliases`, `newaliases`, `~/ .forward`.

Đáp án Bài tập Hướng dẫn

1. Nếu không có thêm tùy chọn hoặc đối số, lệnh `mail henry@lab3.campus` sẽ chuyển sang chế độ nhập để người dùng có thể nhập thư gửi tới `henry@lab3.campus`. Sau khi soạn xong thư, phím nào sẽ đóng chế độ nhập và gửi email đi?

Nhấn `Ctrl` + `D` sẽ khiến chương trình đóng và gửi email.

2. Siêu người dùng có thể thực thi lệnh nào để liệt kê các thư chưa được gửi bắt nguồn từ hệ thống cục bộ?

Lệnh `mailq` hoặc `sendmail -bp`.

3. Làm cách nào để một người dùng không có đặc quyền có thể sử dụng phương thức MTA tiêu chuẩn để tự động chuyển tiếp tất cả thư đến của họ tới địa chỉ `dave@lab2.campus`?

Người dùng nên thêm `dave@lab2.campus` vào `~/ .forward`.

Đáp án Bài tập Mở rộng

1. Bằng cách sử dụng lệnh `mail` do `mailx` cung cấp, lệnh nào sẽ gửi thư đến `emma@lab1.campus` với tệp `logs.tar.gz` dưới dạng tệp đính kèm và đầu ra của lệnh `uname -a` làm nội dung email?

```
uname -a | mail -a logs.tar.gz emma@lab1.campus
```

2. Một quản trị viên dịch vụ email muốn giám sát việc lưu chuyển email qua mạng nhưng họ không muốn làm hộp thư của mình lộn xộn với các thư thử nghiệm. Làm cách nào để quản trị viên này có thể định cấu hình bí danh email trên toàn hệ thống để chuyển hướng tất cả email được gửi tới người dùng `test` sang tệp `/dev/null`?

Dòng `test: /dev/null` trong `/etc/aliases` sẽ chuyển hướng tất cả các thư được gửi đến hộp thư cục bộ `test` sang tệp `/dev/null`.

3. Ngoài `newaliases`, lệnh nào có thể được sử dụng để cập nhật cơ sở dữ liệu bí danh sau khi thêm bí danh mới vào `/etc/aliases`?

Lệnh `sendmail -bi` hoặc `sendmail -I`.



108.4 Quản lý Máy in và tác vụ In

Tham khảo các mục tiêu LPI

[LPIC-1 version 5.0, Exam 102, Objective 108.4](#)

Khối lượng

2

Các lĩnh vực kiến thức chính

- Cấu hình CUPS cơ bản (dành cho máy in cục bộ và từ xa).
- Quản lý hàng đợi in của người dùng.
- Khắc phục các sự cố in ấn chung.
- Thêm và xóa công việc khỏi hàng đợi máy in đã định cấu hình.

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- Các tệp, công cụ và tiện ích cấu hình của CUPS
- `/etc/cups/`
- giao diện kế thừa lpd (`lpr`, `lprm`, `lpq`)



108.4 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	108 Dịch vụ Hệ thống thiết yếu
Mục tiêu:	108.4 Quản lý Máy in và tác vụ In
Bài:	1 trên 1

Giới thiệu

Tính đến thời điểm hiện tại, những tuyên bố về một "xã hội không cần tới giấy tờ" từ sự ra đời của máy tính đã được chứng minh là hoàn toàn sai lầm. Rất nhiều tổ chức vẫn phải phụ thuộc vào các trang thông tin vật lý được in ra hay còn gọi là "bản cứng". Với suy nghĩ này, chúng ta có thể thấy được tầm quan trọng của việc người dùng máy tính biết cách in từ hệ thống cũng như quản trị viên cần biết cách duy trì khả năng làm việc với máy in của máy tính.

Cũng như nhiều hệ điều hành khác, trên Linux, chông phần mềm *Hệ thống In Unix chung* (CUPS) cho phép người dùng in và quản lý máy in từ máy tính. Dưới đây là một bản phác thảo rất đơn giản về cách in một tệp trong Linux bằng CUPS:

1. Người dùng chọn một tệp để in.
2. Trình nền CUPS là `cupsd` sau đó sẽ *cuốn* lệnh in. Lệnh in này sẽ được CUPS cấp một mã số lệnh in, thông tin về hàng đợi in nào sẽ phụ trách lệnh in đó cũng như tên của tài liệu cần in.
3. CUPS sử dụng *các bộ lọc* được cài đặt trên hệ thống để tạo ra các tệp ở định dạng máy in có thể sử dụng được.
4. CUPS sau đó sẽ gửi tệp được định dạng lại tới máy in để in.

Chúng ta sẽ xem xét các bước này cũng như cách cài đặt và quản lý máy in trong Linux một cách chi tiết hơn.

Dịch vụ CUPS

Hầu hết các cài đặt máy tính Linux đều sẽ cài đặt sẵn các gói CUPS. Trên các bản cài đặt Linux tối giản, tùy thuộc vào bản phân phối mà các gói CUPS có thể sẽ không được cài đặt. Việc cài đặt CUPS cơ bản có thể được thực hiện trên hệ thống Debian bằng cách sau:

```
$ sudo apt install cups
```

Trên hệ thống Fedora, quá trình cài đặt cũng rất dễ dàng. Chúng ta sẽ phải khởi động dịch vụ CUPS theo cách thủ công sau khi cài đặt trên Fedora và các bản phân phối dựa trên Red Hat khác:

```
$ sudo dnf install cups
...
$ sudo systemctl start cups.service
```

Sau khi quá trình cài đặt hoàn tất, chúng ta có thể xác minh rằng dịch vụ CUPS đang chạy bằng cách sử dụng lệnh `systemctl`:

```
$ systemctl status cups.service
● cups.service - CUPS Scheduler
   Loaded: loaded (/lib/systemd/system/cups.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2020-06-25 14:35:47 EDT; 41min ago
     Docs: man:cupsd(8)
  Main PID: 3136 (cupsd)
    Tasks: 2 (limit: 1119)
   Memory: 3.2M
   CGroup: /system.slice/cups.service
           └─3136 /usr/sbin/cupsd -l
             └─3175 /usr/lib/cups/notifier/dbus dbus://
```

Giống như nhiều trình nền Linux khác, CUPS cũng phụ thuộc vào một tập hợp các tệp cấu hình để hoạt động. Dưới đây là những vấn đề chính cần được quản trị viên hệ thống quan tâm:

`/etc/cups/cupsd.conf`

Tệp này có chứa các cài đặt cấu hình cho chính dịch vụ CUPS. Nếu đã hoàn toàn quen thuộc với tệp cấu hình máy chủ web Apache thì người dùng cũng sẽ thấy tệp cấu hình CUPS khá quen

thuộc vì nó cũng sử dụng một cú pháp tương đồng. Tệp `cupsd.conf` có chứa các cài đặt cho những nội dung như kiểm soát quyền truy cập vào các hàng đợi in đang được sử dụng trên hệ thống, xem giao diện web CUPS có được bật hay không cũng như mức độ ghi nhật ký mà trình nền sẽ sử dụng.

/etc/printcap

Đây là tệp kế thừa được sử dụng bởi giao thức LPD (*Line Printer Daemon* - Trình nền Máy in Dòng) trước khi CUPS ra đời. CUPS vẫn sẽ tạo tệp này trên các hệ thống để tương thích ngược và nó thường là một liên kết tượng trưng đến `/run/cups/printcap`. Mỗi dòng trong tệp này sẽ chứa một máy in mà hệ thống có quyền truy cập.

/etc/cups/printers.conf

Tệp này có chứa từng máy in được cấu hình để hệ thống CUPS sử dụng. Mỗi máy in và hàng đợi in liên quan của nó trong tệp này được đặt trong một phần `<Printer></Printer>`. Tệp này cung cấp các danh sách máy in riêng lẻ được tìm thấy trong `/etc/printcap`.

WARNING

Đừng thực hiện sửa đổi nào đối với tệp `/etc/cups/printers.conf` tại dòng lệnh trong khi dịch vụ CUPS đang chạy.

/etc/cups/ppd/

Đây không phải là một tệp cấu hình mà là một thư mục chứa các tệp *Mô tả máy in PostScript* (PPD) cho các máy in sử dụng chúng. Khả năng vận hành của mỗi máy in sẽ được lưu trữ trong tệp PPD (kết thúc bằng phần mở rộng `.ppd`). Đây là những tệp văn bản thuần túy và sẽ tuân theo một định dạng cụ thể.

Dịch vụ CUPS cũng sử dụng tính năng ghi nhật ký theo cách tương tự như dịch vụ Apache 2. Nhật ký sẽ được lưu trữ trong `/var/log/cups/` và có chứa `access_log`, `page_log` và `error_log`. `access_log` lưu giữ bản ghi quyền truy cập vào giao diện web CUPS cũng như các hành động được thực hiện trong đó (chẳng hạn như quản lý máy in). `page_log` sẽ theo dõi các lệnh in đã được gửi tới hàng đợi in do cài đặt CUPS quản lý. `error_log` sẽ chứa các thông báo về lệnh in không thành công và các lỗi khác được giao diện web ghi lại.

Tiếp theo, chúng ta sẽ xem xét các công cụ và tiện ích được sử dụng để quản lý dịch vụ CUPS.

Sử dụng Giao diện Web

Như đã nêu ở trên, tệp cấu hình `/etc/cups/cupsd.conf` sẽ xác định xem giao diện web cho hệ thống CUPS có được kích hoạt hay không. Tùy chọn cấu hình sẽ trông giống như sau:

```
# Web interface setting...
```



```
WebInterface Yes
```

Nếu giao diện web được kích hoạt thì CUPS có thể được quản lý từ trình duyệt tại URL mặc định là `http://localhost:631`. Theo mặc định, người dùng trên hệ thống có thể xem máy in và hàng đợi in nhưng mọi hình thức sửa đổi cấu hình đều sẽ yêu cầu người dùng có quyền gốc để xác thực bằng dịch vụ web. Đoạn cấu hình trong tệp `/etc/cups/cupsd.conf` với mục đích hạn chế quyền truy cập vào các khả năng quản trị sẽ trông giống như sau:

```
# All administration operations require an administrator to authenticate...
<Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-Delete-Class
CUPS-Set-Default>
  AuthType Default
  Require user @SYSTEM
  Order deny,allow
</Limit>
```

Hãy cùng phân tích các tùy chọn này dưới đây:

AuthType Default

sẽ sử dụng dấu nhắc xác thực cơ bản khi một hành động yêu cầu quyền truy cập gốc.

Require user @SYSTEM

cho biết rằng người dùng có đặc quyền quản trị sẽ được yêu cầu cho hoạt động. Tùy chọn này có thể được thay đổi thành `@groupname` mà trong đó, các thành viên của `groupname` có thể quản lý dịch vụ CUPS hoặc một danh sách những người dùng cá nhân cụ thể (như trong `Require user carol, tim`) sẽ được cung cấp.

Order deny,allow

hoạt động giống như tùy chọn cấu hình Apache 2 mà trong đó, hành động sẽ bị từ chối theo mặc định trừ khi người dùng (hoặc thành viên của một nhóm) được xác thực.

Giao diện web cho CUPS có thể được vô hiệu hoá bằng cách dừng dịch vụ CUPS, thay đổi tùy chọn `WebInterface` từ `Yes` thành `No` và sau đó khởi động lại dịch vụ CUPS.

Giao diện web CUPS được xây dựng giống như một trang web cơ bản với các tab điều hướng cho các phần khác nhau của hệ thống CUPS. Các tab trong giao diện web bao gồm:

Trang chủ (Home)

Trang chủ sẽ liệt kê phiên bản CUPS hiện tại được cài đặt. Nó cũng sẽ chia CUPS thành các phần như:

CUPS dành cho Người dùng (CUPS for Users)

Cung cấp một mô tả về CUPS, các tùy chọn dòng lệnh để làm việc với máy in và hàng đợi in cũng như một liên kết đến diễn đàn người dùng CUPS.

CUPS dành cho Quản trị viên (CUPS for Administrators)

Cung cấp các liên kết trong giao diện để cài đặt và quản lý máy in cũng như các liên kết đến thông tin về cách làm việc với các máy in trên mạng.

CUPS dành cho Nhà phát triển (CUPS for Developers)

Cung cấp liên kết đến phần phát triển cho chính CUPS cũng như để tạo tệp PPD cho máy in.

Quản trị (Administration)

Trang quản trị cũng được chia thành các phần:

Máy in (Printers)

Tại đây, quản trị viên có thể thêm máy in mới vào hệ thống, định vị các máy in được kết nối với hệ thống và quản lý các máy in đã được cài đặt.

Hạng (Classes)

Hạng là một cơ chế mà trong đó máy in có thể được thêm vào các nhóm với các chính sách cụ thể. Ví dụ: một hạng có thể chứa một nhóm máy in thuộc một tầng cụ thể của một tòa nhà mà chỉ những người dùng trong một bộ phận cụ thể mới có thể in được. Một hạng khác có thể có giới hạn về số lượng trang mà người dùng có thể in. Các hạng sẽ không được tạo theo mặc định khi cài đặt CUPS và phải được quản trị viên xác định. Đây là một phần trong giao diện web CUPS nơi các hạng mới có thể được tạo và quản lý.

Lệnh in (Jobs)

Đây là nơi quản trị viên có thể xem tất cả các lệnh in hiện đang xếp hàng cho tất cả các máy in mà bản cài đặt CUPS này quản lý.

Máy chủ (Server)

Đây là nơi quản trị viên có thể thực hiện các thay đổi đối với tệp `/etc/cups/cupsd.conf`. Ngoài ra, các tùy chọn cấu hình khác cũng có sẵn thông qua các hộp kiểm như cho phép chia sẻ máy in được kết nối với cài đặt CUPS này trên mạng, xác thực nâng cao và cho phép quản trị máy in từ xa.

Hạng (Classes)

Nếu các hạng máy in được cấu hình trên hệ thống, chúng sẽ được liệt kê ở trang này. Mỗi hạng máy in sẽ có các tùy chọn để quản lý tất cả các máy in trong hạng cùng một lúc cũng như xem tất cả các công việc đang chờ cho các máy in trong hạng này.

Trợ giúp (Help)

Tab này sẽ cung cấp các liên kết cho tất cả tài liệu có sẵn về CUPS được cài đặt trên hệ thống.

Công việc (Jobs)

Tab Công việc cho phép người dùng tìm kiếm các lệnh in riêng lẻ cũng như liệt kê tất cả các lệnh in hiện tại do máy chủ quản lý.

Máy in (Printers)

Tab Máy in sẽ liệt kê tất cả các máy in hiện được hệ thống quản lý cũng như một mô tả tổng quan ngắn về trạng thái của từng máy in. Mỗi máy in được liệt kê có thể được nhấp vào và quản trị viên sẽ được đưa đến trang nơi có thể quản lý rộng hơn từng máy in riêng lẻ. Thông tin về các máy in trên tab này đến từ tệp `/etc/cups/printers.conf`.

Cài đặt một Máy in

Việc thêm một hàng đợi máy in vào hệ thống là một tiến trình đơn giản trong giao diện web CUPS:

1. Nhấp vào tab **Administration** rồi nhấp vào nút **Add Printer** (Thêm máy in).
2. Trang tiếp theo sẽ cung cấp nhiều tùy chọn khác nhau tùy thuộc vào cách máy in được kết nối với hệ thống của người dùng. Nếu đó là máy in cục bộ, hãy chọn tùy chọn phù hợp nhất (chẳng hạn như máy in sẽ được kết nối vào cổng nào hoặc phần mềm máy in của bên thứ ba nào có thể được cài đặt). CUPS cũng sẽ cố gắng phát hiện các máy in được kết nối với mạng và hiển thị các máy in đó ở đây. Người dùng cũng có thể chọn tùy chọn kết nối trực tiếp với máy in kết nối mạng tùy thuộc vào giao thức in kết nối mạng mà máy in hỗ trợ. Sau khi chọn tùy chọn thích hợp, hãy nhấp vào nút **Continue** (Tiếp tục).
3. Trang tiếp theo sẽ cho phép người dùng cung cấp tên, mô tả và vị trí (chẳng hạn như “back office” hoặc “front desk”, v.v.) cho máy in. Nếu muốn chia sẻ máy in này qua mạng, người dùng cũng có thể chọn hộp kiểm cho tùy chọn đó trên trang này. Khi cài đặt đã được nhập, hãy nhấp vào nút **Continue**.
4. Trang tiếp theo là nơi người dùng có thể chọn nhãn hiệu và kiểu máy in. Điều này cho phép CUPS tìm kiếm cơ sở dữ liệu được cài đặt cục bộ để tìm trình điều khiển và tệp PPD phù hợp nhất để sử dụng với máy in. Nếu có tệp PPD do nhà sản xuất máy in cung cấp, hãy duyệt đến vị trí của tệp đó và chọn nó để sử dụng tại đây. Sau khi hoàn tất, hãy nhấp vào nút **Add Printer**.
5. Trang cuối cùng là nơi người dùng đặt các tùy chọn mặc định như kích thước trang mà máy in sẽ sử dụng và độ phân giải của các ký tự được in trên trang. Sau khi nhấp vào nút **Set Default Options** (Đặt tùy chọn mặc định), máy in hiện đã được cài đặt trên hệ thống.

NOTE

Nhiều bản cài đặt máy tính của Linux sẽ có các công cụ khác nhau có thể được sử dụng để cài đặt máy in. Môi trường máy tính GNOME và KDE có sẵn các ứng dụng

riêng có thể được sử dụng để cài đặt và quản lý máy in. Ngoài ra, một số bản phân phối còn cung cấp các ứng dụng quản lý máy in riêng biệt. Tuy nhiên, khi xử lý việc cài đặt máy chủ mà nhiều người dùng sẽ in tới, giao diện web CUPS có thể cung cấp những công cụ tốt nhất cho tác vụ này.

Hàng đợi của máy in cũng có thể được cài đặt bằng các lệnh LPD/LPR cũ. Sau đây là một ví dụ sử dụng lệnh `lpadmin`:

```
$ sudo lpadmin -p ENVY-4510 -L "office" -v socket://192.168.150.25 -m everywhere
```

Chúng ta sẽ chia nhỏ lệnh để minh họa các tùy chọn được sử dụng ở đây:

- Vì việc thêm máy in vào hệ thống yêu cầu người dùng có đặc quyền quản trị nên chúng ta sẽ thêm `sudo` vào trước lệnh `lpadmin`.
- Tùy chọn `-p` là đích đến cho lệnh in. Về cơ bản, nó là một cái tên khá thân thiện để người dùng biết các lệnh in sẽ đi đến đâu. Thông thường, chúng ta có thể cung cấp tên của máy in.
- Tùy chọn `-L` là vị trí của máy in. Đây là tùy chọn không bắt buộc nhưng sẽ hữu ích nếu người dùng cần quản lý một số máy in ở nhiều địa điểm khác nhau.
- Tùy chọn `-v` dành cho URI của thiết bị máy in. URI thiết bị là thứ mà hàng đợi in CUPS cần để gửi các lệnh in được kết xuất tới một máy in cụ thể. Trong ví dụ, chúng ta đang sử dụng vị trí mạng bằng địa chỉ IP được cung cấp.
- Tùy chọn cuối cùng, `-m`, được đặt thành “everywhere” (mọi nơi). Tùy chọn này sẽ đặt kiểu máy in cho CUPS để xác định tệp PPD nào sẽ sử dụng. Trong các phiên bản CUPS hiện đại, tốt nhất là chúng ta nên sử dụng “where” để CUPS có thể kiểm tra URI thiết bị (được đặt bằng tùy chọn `-v` trước đó) để tự động xác định đúng tệp PPD cho máy in. Trong các tình huống hiện đại, CUPS sẽ chỉ sử dụng IPP như được giải thích bên dưới.

Như đã nêu trước đây, tốt nhất là chúng ta để CUPS tự động xác định tệp PPD nào sẽ sử dụng cho hàng đợi in cụ thể. Tuy nhiên, lệnh `lpinfo` kế thừa có thể được sử dụng để truy vấn các tệp PPD được cài đặt cục bộ để xem những gì đang có sẵn. Ta chỉ cần cung cấp tùy chọn `--make-and-model` cho máy in cần cài đặt và tùy chọn `-m`:

```
$ lpinfo --make-and-model "HP Envy 4510" -m
hplip:0/ppd/hplip/HP/hp-envy_4510_series-hpijs.ppd HP Envy 4510 Series hpijs, 3.17.10
hplip:1/ppd/hplip/HP/hp-envy_4510_series-hpijs.ppd HP Envy 4510 Series hpijs, 3.17.10
hplip:2/ppd/hplip/HP/hp-envy_4510_series-hpijs.ppd HP Envy 4510 Series hpijs, 3.17.10
drv:///hpcups.crv/hp-envy_4510_series.ppd HP Envy 4510 Series, hpcups 3.17.10
everywhere IPP Everywhere
```

Hãy lưu ý rằng lệnh `lpinfo` không còn được sử dụng nữa. Nó được hiển thị ở đây như một ví dụ về việc liệt kê các tệp trình điều khiển in mà một máy in có thể sử dụng.

WARNING

Các phiên bản tương lai của CUPS không còn dùng trình điều khiển nữa và thay vào đó sẽ tập trung vào việc sử dụng IPP (*Giao thức in Internet*) và các định dạng tệp tiêu chuẩn. Đầu ra của lệnh trước sẽ minh họa điều này bằng khả năng in `where IPP Everywhere`. IPP có thể thực hiện các tác vụ tương tự như trình điều khiển in được sử dụng. IPP, giống như giao diện web CUPS, sử dụng cổng mạng 631 với giao thức TCP.

Chúng ta có thể đặt máy in mặc định bằng lệnh `lpoptions`. Bằng cách này, nếu phần lớn (hoặc tất cả) các lệnh in đều được gửi đến một máy in cụ thể thì máy in được chỉ định bằng lệnh `lpoptions` sẽ là mặc định. Chúng ta chỉ cần chỉ định máy in cùng với tùy chọn `-d`:

```
$ lpoptions -d ENVY-4510
```

Quản lý Máy in

Khi máy in đã được cài đặt, quản trị viên có thể sử dụng giao diện web để quản lý các tùy chọn có sẵn cho máy in. Một cách tiếp cận trực tiếp hơn để quản lý máy in là sử dụng lệnh `lpadmin`.

Có một tùy chọn cho phép một máy in có khả năng được chia sẻ ở bên trong mạng. Điều này có thể đạt được bằng tùy chọn `printer-is-shared` và chỉ định máy in với tùy chọn `-p`:

```
$ sudo lpadmin -p FRONT-DESK -o printer-is-shared=true
```

Quản trị viên cũng có thể định cấu hình hàng đợi in để chỉ chấp nhận lệnh in từ những người dùng cụ thể với mỗi người dùng được phân tách bằng một dấu phẩy:

```
$ sudo lpadmin -p FRONT-DESK -u allow:carol,frank,grace
```

Ngược lại, chỉ những người dùng cụ thể mới có thể bị từ chối truy cập vào hàng đợi in cụ thể:

```
$ sudo lpadmin -p FRONT-DESK -u deny:dave
```

Các nhóm người dùng cũng có thể được sử dụng để cho phép hoặc từ chối quyền truy cập vào hàng đợi của máy in với điều kiện tên của nhóm được đặt trước ký tự “at” (@):

```
$ sudo lpadmin -p FRONT-DESK -u deny:@sales,@marketing
```

Hàng đợi in cũng có thể có chính sách lỗi nếu nó gặp phải sự cố khi in. Với việc sử dụng các chính sách, một lệnh in có thể bị hủy bỏ (`abort-job`) hoặc một tác vụ in khác có thể xảy ra tại một thời điểm sau đó (`retry-job`). Các chính sách khác bao gồm khả năng dừng máy in ngay lập tức nếu xảy ra lỗi (`stop-printer`) cũng như khả năng thử lại lệnh in ngay sau khi phát hiện lỗi (`retry-current-job`). Dưới đây là một ví dụ mà trong đó, chính sách máy in được đặt để hủy lệnh in nếu xảy ra lỗi trên máy in FRONT-DESK:

```
$ sudo lpadmin -p FRONT-DESK -o printer-error-policy=abort-job
```

Hãy xem lại các trang hướng dẫn của lệnh `lpadmin` tại `lpadmin(8)` để biết thêm chi tiết về cách sử dụng lệnh này.

Gửi Lệnh In

Nhiều ứng dụng máy tính sẽ cho phép người dùng gửi lệnh in từ một mục menu hoặc bằng cách sử dụng phím tắt `Ctrl + p`. Nếu sử dụng hệ thống Linux không sử dụng môi trường máy tính, chúng ta vẫn có thể gửi tệp tới máy in bằng các lệnh LPD/LPR cũ.

Lệnh `lpr` (“line printer remote”) được sử dụng để gửi lệnh in đến hàng đợi của máy in. Ở dạng cơ bản nhất của lệnh, tên tệp cùng với lệnh `lpr` là tất cả những gì chúng ta cần:

```
$ lpr report.txt
```

Lệnh trên sẽ gửi tệp `report.txt` đến hàng đợi in mặc định cho hệ thống (như được xác định bởi tệp `/etc/cups/printers.conf`).

Nếu một cài đặt CUPS có nhiều máy in được cài đặt thì lệnh `lpstat` có thể được sử dụng với tùy chọn `-p` để in ra một danh sách các máy in khả dụng và tùy chọn `-d` sẽ cho biết đâu là máy in mặc định:

```
$ lpstat -p -d
printer FRONT-DESK is idle.  enabled since Mon 03 Aug 2020 10:33:07 AM EDT
printer PostScript_oc0303387803 disabled since Sat 07 Mar 2020 08:33:11 PM EST -
    reason unknown
printer ENVY-4510 is idle.  enabled since Fri 31 Jul 2020 10:08:31 AM EDT
system default destination: ENVY-4510
```

Vì vậy, trong ví dụ của chúng ta, tệp `report.txt` sẽ được gửi đến máy in ENVY-4510 vì nó được đặt làm mặc định. Nếu tệp cần được in ở một máy in khác, hãy chỉ định máy in cùng với tùy chọn `-P`:

```
$ lpr -P FRONT-DESK report.txt
```

Khi lệnh in được gửi tới CUPS, trình nền này sẽ tìm ra phần liên quan nào phù hợp nhất để xử lý tác vụ. CUPS có thể sử dụng nhiều trình điều khiển máy in, bộ lọc, màn hình cổng kết nối phần cứng và các phần mềm khác để hiển thị tài liệu một cách chính xác. Sẽ có lúc người dùng cần in một tài liệu nơi cách thức in cần phải sửa đổi. Nhiều ứng dụng đồ họa có thể thực hiện nhiệm vụ này một cách khá dễ dàng. Ngoài ra, chúng ta còn có các tùy chọn dòng lệnh có thể được sử dụng để thay đổi cách in tài liệu. Khi lệnh in được gửi qua dòng lệnh, khoá chuyển `-o` (“options”) có thể được sử dụng cùng với các thuật ngữ cụ thể để điều chỉnh bố cục của tài liệu khi in. Dưới đây là một danh sách ngắn các tùy chọn thường được sử dụng:

landscape

Tài liệu được in dưới dạng trang được xoay 90 độ theo chiều kim đồng hồ. Tùy chọn `orientation-requested=4` cũng sẽ đạt được kết quả tương tự.

two-sided-long-edge

Máy in sẽ in tài liệu ở chế độ dọc trên cả hai mặt giấy (trong trường hợp máy in có hỗ trợ khả năng này).

two-sided-short-edge

Máy in sẽ in tài liệu ở chế độ ngang trên cả hai mặt giấy (trong trường hợp máy in có hỗ trợ khả năng này).

media

Máy in sẽ in lệnh in theo kích thước phương tiện được chỉ định. Kích thước phương tiện có sẵn cho lệnh in sử dụng sẽ tùy thuộc vào máy in, nhưng dưới đây là một danh sách các kích thước phổ biến:

Tùy chọn kích thước	Ý nghĩa
A4	ISO A4
Letter	Thư tiêu chuẩn Mỹ (US Letter)
Legal	Pháp lý tiêu chuẩn Mỹ (US Legal)
DL	Phong bì ISO DL (ISO DL Envelope)
COM10	Phong bì tiêu chuẩn Mỹ #10 (US #10 Envelope)

collate

sắp xếp tài liệu được in. Tùy chọn này rất hữu ích nếu người dùng có một tài liệu nhiều trang cần được in nhiều lần vì khi đó, tất cả các trang cho mỗi tài liệu sẽ được in theo thứ tự. Đặt tùy chọn này thành `true` để kích hoạt hoặc `false` để vô hiệu hoá nó.

page-ranges

Tùy chọn này có thể được sử dụng để chọn một trang để in hoặc một tập hợp các trang cụ thể để in từ tài liệu. Một ví dụ sẽ như sau: `-o page-ranges=5-7,9,15`. Lệnh này sẽ in các trang 5, 6 và 7 sau đó là trang 9 và 15.

fit-to-page

In tài liệu sao cho tệp cân tỷ lệ vừa với giấy. Nếu tệp cần in không cung cấp thông tin về kích thước trang thì có thể lệnh in sẽ bị thu nhỏ một cách không chính xác và các phần của tài liệu có thể bị sai tỉ lệ trong trang hoặc bị thu lại còn rất nhỏ.

outputorder

In tài liệu theo chiều nghịch (`reverse`) hoặc chiều thuận (`normal`) để bắt đầu in trên trang đầu tiên. Nếu một máy in in các trang theo chiều úp xuống, thứ tự mặc định sẽ là `-o outputorder=normal` trong khi các máy in in với các trang theo chiều hướng lên trên sẽ in với `-o outputorder=reverse`.

Lấy mẫu từ các tùy chọn ở trên, chúng ta có thể xây dựng lệnh ví dụ như sau:

```
$ lpr -P ACCOUNTING-LASERJET -o landscape -o media=A4 -o two-sided-short-edge finance-report.pdf
```

Chúng ta có thể in nhiều bản sao của một tài liệu bằng cách sử dụng tùy chọn số ở định dạng sau: `-#N`, trong đó, `N` bằng với số lượng bản sao cần in. Đây là một ví dụ với tùy chọn `collate` mà trong đó bảy bản sao của một báo cáo sẽ được in trên máy in mặc định:

```
$ lpr -#7 -o collate=true status-report.pdf
```

Ngoài lệnh `lpr`, lệnh `lp` cũng có thể được sử dụng. Nhiều tùy chọn được sử dụng với lệnh `lpr` cũng có thể được sử dụng với lệnh `lp` nhưng sẽ có một số khác biệt. Hãy tham khảo trang hướng dẫn tại `lp(1)` để nắm rõ hơn. Dưới đây sẽ là cách chúng ta có thể chạy lệnh `lpr` trong ví dụ trước bằng cách sử dụng cú pháp của lệnh `lp`, đồng thời chỉ định máy in đích với tùy chọn `-d`:

```
$ lp -d ACCOUNTING-LASERJET -n 7 -o collate=true status-report.pdf
```


Quản lý Lệnh In

Như đã nêu trước đó, mỗi lệnh in được gửi tới hàng đợi in sẽ nhận được một ID công việc từ CUPS. Người dùng có thể xem các lệnh in mà họ đã gửi bằng lệnh `lpq`. Việc truyền tùy chọn `-a` sẽ hiển thị hàng đợi của tất cả các máy in được quản lý bởi cài đặt CUPS:

```
$ lpq -a
Rank      Owner      Job      File(s)      Total Size
1st       carol      20       finance-report.pdf  5072 bytes
```

Lệnh `lpstat` tương tự được sử dụng trước đây cũng có tùy chọn để xem hàng đợi máy in. Bản thân tùy chọn `-o` sẽ hiển thị tất cả các hàng đợi in hoặc hàng đợi in có thể được chỉ định theo tên:

```
$ lp -o
ACCOUNTING-LASERJET-4          carol      19456   Wed 05 Aug 2020 04:29:44 PM EDT
```

ID lệnh in sẽ được thêm vào sau tên của hàng đợi nơi lệnh in được gửi, sau đó là tên của người dùng đã gửi lệnh in, kích thước tệp và thời gian lệnh in được gửi.

Nếu lệnh in bị kẹt trên máy in hoặc người dùng muốn hủy lệnh in của họ, hãy sử dụng lệnh `lprm` cùng với ID công việc căn cứ từ lệnh `lpq`:

```
$ lprm 20
```

Tất cả các công việc trong hàng đợi in có thể bị xóa cùng một lúc bằng cách chỉ cung cấp dấu gạch ngang `-`:

```
$ lprm -
```

Ngoài ra, người dùng cũng có thể sử dụng lệnh CUPS `cancel` để dừng lệnh in hiện tại của họ:

```
$ cancel
```

Một lệnh in cụ thể có thể bị hủy bằng ID lệnh in được thêm vào sau tên máy in:

```
$ cancel ACCOUNTING-LASERJET-20
```

Lệnh in cũng có thể được chuyển từ hàng đợi in này sang hàng đợi in khác. Điều này thường sẽ hữu ích khi máy in ngừng phản hồi hoặc tài liệu được in yêu cầu các tính năng có sẵn trên một máy in khác. Hãy lưu ý rằng quy trình này thường yêu cầu người dùng có đặc quyền nâng cao. Bằng cách sử dụng cùng một lệnh in từ ví dụ trước, chúng ta có thể di chuyển nó đến hàng đợi của máy in FRONT-DESK:

```
$ sudo lpmove ACCOUNTING-LASERJET-20 FRONT-DESK
```

Xóa máy in

Để xóa một máy in, trước tiên chúng ta nên liệt kê tất cả các máy in hiện được dịch vụ CUPS quản lý. Điều này có thể được thực hiện bằng lệnh `lpstat`:

```
$ lpstat -v
device for FRONT-DESK: socket://192.168.150.24
device for ENVY-4510: socket://192.168.150.25
device for PostScript_oc0303387803: ///dev/null
```

Tùy chọn `-v` không chỉ liệt kê các máy in mà còn cả vị trí (và cách thức) nơi chúng được gắn vào. Cách tốt nhất là trước tiên chúng ta sẽ từ chối bất kỳ một lệnh in mới nào được gửi tới máy in và đưa ra lý do tại sao máy in sẽ không nhận lệnh in mới. Điều này có thể được thực hiện bằng cách sau:

```
$ sudo cupsreject -r "Printer to be removed" FRONT-DESK
```

Hãy lưu ý việc sử dụng `sudo` vì tác vụ này yêu cầu người dùng có đặc quyền nâng cao.

Để xóa một máy in, chúng ta có thể sử dụng lệnh `lpadmin` với tùy chọn `-x`:

```
$ sudo lpadmin -x FRONT-DESK
```

Bài tập Hướng dẫn

1. Một máy in mới vừa được cài đặt trên máy trạm cục bộ có tên `office-mgr`. Lệnh nào có thể được sử dụng để đặt máy in này làm mặc định cho máy trạm này?
2. Lệnh và tùy chọn nào sẽ được sử dụng để xác định máy in nào có sẵn để in từ máy trạm?
3. Bằng cách sử dụng lệnh `cancel`, bạn sẽ xóa lệnh in có ID 15 bị kẹt trong hàng đợi của máy in có tên là `office-mgr` như thế nào?
4. Bạn có một lệnh in dành cho một máy in không có đủ giấy để in toàn bộ tệp. Bạn sẽ sử dụng lệnh nào để di chuyển lệnh in có ID 2 được xếp hàng đợi in trên máy in `FRONT-DESK` sang hàng đợi in cho máy in `ACCOUNTING-LASERJET`?

Bài tập Mở rộng

Bằng cách sử dụng trình quản lý gói của bản phân phối của bạn, hãy cài đặt các gói `cups` và `printer-driver-cups-pdf`. Hãy lưu ý rằng nếu bạn đang sử dụng bản phân phối dựa trên Red Hat (chẳng hạn như Fedora) thì trình điều khiển CUPS PDF sẽ được gọi là `cups-pdf`. Hãy cài đặt đồng thời cả gói `cups-client` để sử dụng các lệnh in kiểu System V. Chúng ta sẽ sử dụng các gói này để thực hành quản lý máy in CUPS mà không cần cài đặt vật lý một máy in thực.

1. Hãy xác minh rằng trình nền CUPS đang chạy, sau đó xác minh rằng máy in PDF đã được bật và đặt nó thành mặc định.

2. Hãy chạy lệnh sẽ in tệp `/etc/services`. Bây giờ, bạn sẽ có một thư mục có tên PDF trong thư mục chính của bạn.

3. Hãy sử dụng một lệnh mà chỉ tắt máy in, sau đó chạy một lệnh riêng để hiển thị tất cả các thông tin trạng thái để xác minh rằng máy in PDF đã bị tắt. Sau đó, hãy thử in một bản sao của tệp `/etc/fstab` của bạn. Điều gì sẽ xảy ra?

4. Bây giờ, hãy thử in một bản sao của tệp `/etc/fstab` bằng máy in PDF. Điều gì sẽ xảy ra?

5. Hãy hủy lệnh in, sau đó xoá máy in PDF.

Tóm tắt

Trình nền CUPS là một nền tảng được sử dụng rộng rãi để in dành cho các máy in cục bộ và từ xa. Mặc dù nó được sử dụng để thay thế cho giao thức LPD cũ nhưng nó vẫn cung cấp khả năng tương thích ngược cho các công cụ của mình.

Các tệp và lệnh đã được thảo luận trong bài học này là:

/etc/cups/cupsd.conf

Tệp cấu hình chính cho chính dịch vụ CUPS. Tệp này cũng kiểm soát quyền truy cập vào giao diện web dành cho CUPS.

/etc/printcap

Một tệp kế thừa được LPD sử dụng có chứa các dòng cho mỗi máy in được kết nối với hệ thống.

/etc/cups/printers.conf

Tệp cấu hình được CUPS sử dụng để nắm giữ thông tin máy in.

Bạn có thể tìm thấy giao diện web CUPS trong cài đặt mặc định tại <http://localhost:631>. Hãy nhớ rằng, cổng mạng mặc định cho giao diện web là 631/TCP.

Các lệnh LPD/LPR kế thừa sau đây cũng đã được thảo luận:

lpadmin

Được sử dụng để cài đặt và gỡ bỏ máy in và các hạng máy in.

lpoptions

Được sử dụng để in ra các tùy chọn máy in và sửa đổi cài đặt của máy in.

lpstat

Được sử dụng để hiển thị thông tin trạng thái của máy in được kết nối với cài đặt CUPS.

lpr

Được sử dụng để gửi lệnh in tới hàng đợi của máy in.

lp

Được sử dụng để gửi lệnh in tới hàng đợi của máy in.

lpq

Được sử dụng để liệt kê các lệnh in trong hàng đợi in.

lprm

Được sử dụng để hủy lệnh in theo ID. ID cho một lệnh in có thể được lấy bằng đầu ra của lệnh `lpq`.

cancel

Một thay thế cho lệnh `lprm` để hủy lệnh in bằng ID của chúng.

Hãy nhớ xem lại các trang hướng dẫn sau để biết về các công cụ và tiện ích khác nhau của CUPS: `lpadmin(8)`, `lpoptions(1)`, `lpr(1)`, `lpq(1)`, `lprm(1)`, `cancel(1)`, `lpstat(1)`, `cupsenable(8)` và `cupsaccept(8)`. Bạn cũng nên xem lại tài liệu trợ giúp trực tuyến tại <http://localhost:631/help>.

Đáp án Bài tập Hướng dẫn

1. Một máy in mới vừa được cài đặt trên máy trạm cục bộ có tên `office-mgr`. Lệnh nào có thể được sử dụng để đặt máy in này làm mặc định cho máy trạm này?

```
$ lpoptions -d office-mgr
```

2. Lệnh và tùy chọn nào sẽ được sử dụng để xác định máy in nào có sẵn để in từ máy trạm?

```
$ lpstat -p
```

Tùy chọn `-p` sẽ liệt kê tất cả các máy in có sẵn và liệu chúng có đang được bật để sẵn sàng in hay không.

3. Bằng cách sử dụng lệnh `cancel`, bạn sẽ xóa lệnh in có ID 15 bị kẹt trong hàng đợi của máy in có tên là `office-mgr` như thế nào?

```
$ cancel office-mgr-15
```

4. Bạn có một lệnh in dành cho một máy in không có đủ giấy để in toàn bộ tệp. Bạn sẽ sử dụng lệnh nào để di chuyển lệnh in có ID 2 được xếp hàng đợi in trên máy in `FRONT-DESK` sang hàng đợi in cho máy in `ACCOUNTING-LASERJET`?

```
$ sudo lpmove FRONT-DESK-2 ACCOUNTING-LASERJET
```

Đáp án Bài tập Mở rộng

Bằng cách sử dụng trình quản lý gói của bản phân phối của bạn, hãy cài đặt các gói `cups` và `printer-driver-cups-pdf`. Hãy lưu ý rằng nếu bạn đang sử dụng bản phân phối dựa trên Red Hat (chẳng hạn như Fedora) thì trình điều khiển CUPS PDF sẽ được gọi là `cups-pdf`. Hãy cài đặt đồng thời cả gói `cups-client` để sử dụng các lệnh in kiểu System V. Chúng ta sẽ sử dụng các gói này để thực hành quản lý máy in CUPS mà không cần cài đặt vật lý một máy in thực.

1. Hãy xác minh rằng trình nền CUPS đang chạy, sau đó xác minh rằng máy in PDF đã được bật và đặt nó thành mặc định.

Một phương pháp để kiểm tra tính khả dụng và trạng thái của máy in PDF là chạy lệnh sau:

```
$ lpstat -p -d

printer PDF is idle.  enabled since Thu 25 Jun 2020 02:36:07 PM EDTi

system default destination: PDF
```

2. Hãy chạy lệnh sẽ in tệp `/etc/services`. Bây giờ, bạn sẽ có một thư mục có tên `PDF` trong thư mục chính của bạn.

```
$ lp -d PDF /etc/services
```

sẽ có tác dụng. Bây giờ bạn sẽ có phiên bản PDF của tệp này trong thư mục `PDF`.

3. Hãy sử dụng một lệnh mà chỉ tắt máy in, sau đó chạy một lệnh riêng để hiển thị tất cả các thông tin trạng thái để xác minh rằng máy in PDF đã bị tắt. Sau đó, hãy thử in một bản sao của tệp `/etc/fstab` của bạn. Điều gì sẽ xảy ra?

```
$ sudo cupsdisable PDF
```

sẽ vô hiệu hóa máy in.

Tiếp theo là chạy lệnh `lpstat -t` để có một danh sách đầy đủ về tình trạng của máy in. Nó sẽ trông giống như đầu ra sau:

```
$ scheduler is running
```



```
system default destination: PDFi

device for PDF: cups-pdf:/

PDF accepting requests since Wed 05 Aug 2020 04:19:15 PM EDTi

printer PDF disabled since Wed 05 Aug 2020 04:19:15 PM EDT -

Paused
```

4. Bây giờ, hãy thử in một bản sao của tệp `/etc/fstab` bằng máy in PDF. Điều gì sẽ xảy ra?

Sau khi thử lệnh `lp -d PDF /etc/fstab`, bạn sẽ nhận được kết quả hiển thị thông tin ID lệnh in. Tuy nhiên, nếu bạn kiểm tra thư mục PDF trong thư mục chính thì tệp mới sẽ không có ở đó. Tiếp theo, bạn có thể kiểm tra hàng đợi in bằng lệnh `lpstat -o` và sẽ thấy lệnh in của mình được liệt kê ở đó.

5. Hãy hủy lệnh in, sau đó xoá máy in PDF.

Bằng cách sử dụng đầu ra từ lệnh `lp` trước đó, bạn có thể sử dụng lệnh `cancel` để xoá lệnh in. Ví dụ:

```
$ cancel PDF-4
```

Sau đó chạy lệnh `lpstat -o` để xác minh rằng lệnh in đã bị xoá.

Xoá máy in PDF bằng lệnh `sudo lpadmin -x PDF`, sau đó xác minh rằng máy in đã bị xoá bằng `lpstat -a`.



Chủ đề 109: Nguyên tắc cơ bản về Mạng



109.1 Nguyên tắc cơ bản của các Giao thức Internet

Tham khảo các mục tiêu LPI

[LPIC-1 version 5.0, Exam 102, Objective 109.1](#)

Khối lượng

4

Các lĩnh vực kiến thức chính

- Thể hiện hiểu biết về mặt nạ mạng và ký hiệu CIDR.
- Kiến thức về sự khác biệt giữa các địa chỉ IP "tứ giác chấm" riêng tư và công khai.
- Kiến thức về các cổng và dịch vụ TCP và UDP phổ biến (20, 21, 22, 23, 25, 53, 80, 110, 123, 139, 143, 161, 162, 389, 443, 465, 514, 636, 993, 995) .
- Kiến thức về sự khác biệt và các tính năng chính của UDP, TCP và ICMP.
- Kiến thức về sự khác biệt chính giữa IPv4 và IPv6.
- Kiến thức về các tính năng cơ bản của IPv6.

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- `/etc/services`
- IPv4, IPv6
- Subnetting
- TCP, UDP, ICMP



109.1 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	109 Những yếu tố cơ bản về Mạng
Mục tiêu:	109.1 Những yếu tố cơ bản của các Giao thức Internet
Bài:	1 trên 2

Giới thiệu

TCP/IP (*Giao thức điều khiển truyền tải/Giao thức Internet*) là một tập hợp các giao thức được sử dụng để cho phép các máy tính liên lạc với nhau. Mặc dù có tên ngắn gọn như vậy nhưng tập hợp này bao gồm khá nhiều các giao thức như IP, TCP, UDP, ICMP, DNS, SMTP, ARP, v.v.

IP (Giao thức Internet)

IP là một giao thức chịu trách nhiệm về thiết lập địa chỉ logic của một máy chủ. Nó cho phép gói được gửi từ máy chủ này sang máy chủ khác. Trên phương diện này, mỗi thiết bị trên mạng sẽ được gán một địa chỉ IP duy nhất và một thiết bị có thể được gán nhiều địa chỉ cùng một lúc.

Trong phiên bản thứ 4 của giao thức IP (thường gọi là IPv4), địa chỉ sẽ được hình thành bởi một bộ 32 bit được phân tách thành 4 nhóm 8 bit được biểu diễn dưới dạng thập phân và được gọi là “tứ giác chấm”. Ví dụ:

Định dạng nhị phân (4 nhóm 8 bit)

```
11000000.10101000.00001010.00010100
```

Định dạng thập phân

192.168.10.20

Trong IPv4, các giá trị cho mỗi 8 bit có thể nằm trong khoảng từ 0 đến 255, tương đương với 11111111 ở định dạng nhị phân.

Các Hạng địa chỉ

Về mặt lý thuyết, các địa chỉ IP được phân tách theo các hạng (class) được xác định bởi phạm vi của 8 bit đầu tiên như trong bảng dưới đây:

Hạng	8 bit đầu tiên	Phạm vi	Ví dụ
A	1-126	1.0.0.0 – 126.255.255.255	10.25.13.10
B	128-191	128.0.0.0 – 191.255.255.255	141.150.200.1
C	192-223	192.0.0.0 – 223.255.255.255	200.178.12.242

IP công cộng và riêng tư

Như đã đề cập từ trước, để giao tiếp có thể diễn ra, mỗi một thiết bị trên mạng phải được liên kết với ít nhất một địa chỉ IP của riêng nó. Tuy nhiên, nếu mỗi một thiết bị kết nối Internet trên thế giới đều có một địa chỉ IP duy nhất thì sẽ không có đủ IP (v4) cho tất cả mọi người. Vì lý do này, địa chỉ IP *riêng tư* đã được ra đời.

IP riêng tư là các dải địa chỉ IP được dành riêng để sử dụng trong mạng nội bộ (riêng tư) của các công ty, tổ chức, gia đình, v.v. Trong cùng một mạng, các địa chỉ IP vẫn sẽ là duy nhất. Tuy nhiên, cùng một địa chỉ IP riêng tư có thể được sử dụng trong bất kỳ một mạng riêng tư nào.

Do đó, trên Internet, chúng ta có lưu lượng dữ liệu sử dụng các địa chỉ IP công cộng có thể nhận dạng và định tuyến qua Internet trong khi các mạng riêng sẽ sử dụng các dải IP được đặt riêng. Bộ định tuyến có nhiệm vụ chuyển đổi lưu lượng truy cập từ mạng riêng tư sang mạng công cộng và ngược lại.

Chúng ta có thể xem phạm vi IP riêng tư được phân tách theo hạng trong bảng bên dưới:

Hạng	8 bit đầu tiên	Phạm vi	IP riêng tư
A	1-126	1.0.0.0 – 126.255.255.255	10.0.0.0 – 10.255.255.255
B	128-191	128.0.0.0 – 191.255.255.255	172.16.0.0 – 172.31.255.255
C	192-223	192.0.0.0 – 223.255.255.255	192.168.0.0 – 192.168.255.255

Chuyển đổi từ định dạng Thập phân sang Nhị phân

Đối với đối tượng của chủ đề này, quan trọng nhất là chúng ta phải biết cách chuyển đổi địa chỉ IP giữa định dạng nhị phân và thập phân.

Việc chuyển đổi từ định dạng thập phân sang nhị phân được thực hiện thông qua các phép chia liên tiếp cho 2. Ví dụ: hãy cùng chuyển đổi giá trị 105 theo các bước sau:

1. Chia giá trị 105 cho 2 ta có:

```
105/2
Quotient = 52
Rest = 1
```

2. Chia thương (quotient) tuần tự cho 2 cho đến khi thương bằng 1:

```
52/2
Rest = 0
Quotient = 26
```

```
26/2
Rest = 0
Quotient = 13
```

```
13/2
Rest = 1
Quotient = 6
```

```
6/2
```

```
Rest = 0
Quotient = 3
```

```
2/3
Rest = 1
Quotient = 1
```

Nhóm thương số của phép tính cuối cùng lại với các số dư (rest) của tất cả các phép chia trước đó:

+

```
1101001
```

Điền các số 0 vào bên trái cho đến khi hoàn thiện 8 bit:

+

```
01101001
```

1. Cuối cùng, chúng ta có giá trị 105 ở dạng thập phân sẽ là 01101001 ở dạng nhị phân.

Chuyển đổi từ định dạng Nhị phân sang Thập phân

Trong ví dụ này, chúng ta sẽ sử dụng giá trị nhị phân 10110000.

1. Mỗi bit đều được liên kết với một giá trị của phép lũy thừa cơ số 2. Phép lũy thừa có số mũ được bắt đầu từ 0 và sẽ tăng dần từ phải sang trái. Trong ví dụ này, chúng ta sẽ có:

1	0	1	1	0	0	0	0
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

2. Khi bit bằng 1, chúng ta sẽ gán giá trị lũy thừa tương ứng; khi bit bằng 0 thì kết quả sẽ là 0.

1	0	1	1	0	0	0	0
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	0	32	16	0	0	0	0

3. Cộng tất cả các giá trị lại:

$$128 + 32 + 16 = 176$$

4. Do đó, 10110000 ở dạng nhị phân sẽ là 176 ở dạng thập phân.

Mặt nạ mạng

Mặt nạ mạng (hoặc *netmask*) được sử dụng cùng với địa chỉ IP để xác định phần nào của IP sẽ đại diện cho mạng và phần nào sẽ đại diện cho máy chủ. Nó sẽ có cùng định dạng với địa chỉ IP, tức là có 32 bit chia thành 4 nhóm 8. Ví dụ:

Thập phân	Nhị phân	CIDR
255.0.0.0	11111111.00000000.00000000 0.00000000	8
255.255.0.0	11111111.11111111.00000000 0.00000000	16
255.255.255.0	11111111.11111111.11111111 1.00000000	24

Sử dụng mặt nạ 255.255.0.0 làm ví dụ, nó đã chỉ ra rằng trong IP được liên kết với nó, 16 bit đầu tiên (2 số thập phân đầu tiên) đã xác định mạng/mạng con và 16 bit cuối cùng được sử dụng để nhận dạng duy nhất các máy chủ trong mạng.

CIDR (*Định tuyến vô Hạng giữa các Miền*) được đề cập ở trên có liên quan đến một ký hiệu mặt nạ được đơn giản hoá cho biết số bit (1) được liên kết với mạng/mạng con. Ký hiệu này thường được sử dụng để thay thế định dạng thập phân (ví dụ như /24 thay vì 255.255.255.0).

Điều thú vị cần lưu ý là mỗi hạng IP đều có một mặt nạ tiêu chuẩn như sau:

Hạng	8 bit đầu tiên	Phạm vi	Mặt nạ mặc định
A	1-126	1.0.0.0 – 126.255.255.255	255.0.0.0 / 8
B	128-191	128.0.0.0 – 191.255.255.255	255.255.0.0 / 16
C	192-223	192.0.0.0 – 223.255.255.255	255.255.255.0 / 24

Tuy nhiên, mẫu này không có nghĩa đây là mặt nạ sẽ luôn được sử dụng. Chúng ta có thể sử dụng bất kỳ mặt nạ nào với bất kỳ một địa chỉ IP nào như có thể thấy ở dưới đây.

Dưới đây là một số ví dụ về việc sử dụng IP và Mặt nạ:

```
192.168.8.12 / 255.255.255.0 / 24
```

Range

```
192.168.8.0 - 192.168.8.255
```

Network Address

```
192.168.8.0
```

Broadcast Address

```
192.168.8.255
```

Hosts

```
192.168.8.1 - 192.168.8.254
```

Trong trường hợp này, chúng ta có 3 chữ số đầu tiên (24 bit đầu tiên) của địa chỉ IP xác định mạng và chữ số cuối cùng xác định địa chỉ của máy chủ. Điều này có nghĩa là phạm vi của mạng này đi từ 192.168.8.0 đến 192.168.8.255.

Bây giờ chúng ta đã có hai khái niệm quan trọng: Mỗi mạng/mạng con đều có 2 địa chỉ dành riêng cho nó và địa chỉ đầu tiên trong dải được gọi là *địa chỉ mạng* (network address). Trong trường hợp này, 192.168.8.0 được sử dụng để xác định chính mạng/mạng con. Địa chỉ cuối cùng trong phạm vi được gọi là *địa chỉ truyền phát* (broadcast address), trong trường hợp này là 192.168.8.255. Địa chỉ đích này được sử dụng để gửi cùng một thông báo (gói) tới tất cả các máy chủ IP trên mạng/mạng con đó.

Các máy trên mạng không thể sử dụng địa chỉ mạng và địa chỉ truyền phát. Do đó, danh sách IP có thể được cấu hình hiệu quả nằm trong khoảng từ 192.168.8.1 đến 192.168.8.254.

Sau đây là một ví dụ về cùng một IP nhưng với một mặt nạ khác:

```
192.168.8.12 / 255.255.0.0 / 16
```

Range

```
192.168.0.0 - 192.168.255.255
```

Network Address

```
192.168.0.0
```

Broadcast Address

192.168.255.255

Hosts

192.168.0.1 – 192.168.255.254

Hãy xem cách một mặt nạ khác đã thay đổi phạm vi IP trong cùng một mạng/mạng con như thế nào.

Việc phân chia mạng theo mặt nạ không bị giới hạn ở các giá trị mặc định (8, 16, 24). Chúng ta có thể tạo các phân mục theo ý muốn, thêm hoặc bớt các bit trong nhận dạng mạng hay tạo các mạng con mới.

Ví dụ:

11111111.11111111.11111111.00000000 = 255.255.255.0 = 24

Nếu muốn chia mạng ở trên thành 2, chúng ta chỉ cần thêm một bit khác vào nhận dạng mạng trong mặt nạ như sau:

11111111.11111111.11111111.10000000 = 255.255.255.128 = 25

Khi đó, chúng ta sẽ có các mạng con sau:

192.168.8.0 - 192.168.8.127
192.168.8.128 - 192.168.8.255

Nếu chúng ta tăng thêm mức phân chia cho mạng:

11111111.11111111.11111111.11000000 = 255.255.255.192 = 26

Ta sẽ có:

192.168.8.0 - 192.168.8.63
192.168.8.64 - 192.168.8.127
192.168.8.128 - 192.168.8.191
192.168.8.192 - 192.168.8.255

Hãy lưu ý rằng trong mỗi một mạng con chúng ta đều sẽ có địa chỉ mạng đặt riêng (địa chỉ đầu tiên trong phạm vi) và địa chỉ truyền phát (địa chỉ cuối cùng trong phạm vi). Vì vậy nên mạng càng được chia nhỏ thì các máy chủ càng có ít IP có thể sử dụng hiệu quả hơn.

Xác định Địa chỉ Mạng và Địa chỉ Truyền phát

Thông qua một Địa chỉ IP và một Mặt nạ, chúng ta có thể xác định được địa chỉ mạng và địa chỉ truyền phát, từ đó xác định phạm vi IP cho mạng/mạng con.

Địa chỉ mạng được lấy bằng cách sử dụng một “Logical AND” giữa địa chỉ IP và mặt nạ ở định dạng nhị phân của chúng. Hãy lấy một ví dụ sử dụng IP 192.168.8.12 và mặt nạ 255.255.255.192.

Khi chuyển đổi từ định dạng thập phân sang nhị phân như chúng ta đã thấy trước đó, ta sẽ có:

```
11000000.10101000.00001000.00001100 (192.168.8.12)
11111111.11111111.11111111.11000000 (255.255.255.192)
```

Với “Logical AND”, ta có 1 và 1 = 1, 0 và 0 = 0, 1 và 0 = 0, vậy nên:

```
11000000.10101000.00001000.00001100 (192.168.8.12)
11111111.11111111.11111111.11000000 (255.255.255.192)
11000000.10101000.00001000.00000000
```

Vậy địa chỉ mạng cho mạng con đó sẽ là 192.168.8.0.

Để có được địa chỉ truyền phát, chúng ta phải sử dụng địa chỉ mạng nơi tất cả các bit liên quan đến địa chỉ máy chủ đều là 1:

```
11000000.10101000.00001000.00000000 (192.168.8.0)
11111111.11111111.11111111.11000000 (255.255.255.192)
11000000.10101000.00001000.00111111
```

Địa chỉ truyền phát khi đó sẽ là 192.168.8.63.

Tóm lại, chúng ta có:

```
192.168.8.12 / 255.255.255.192 / 26
```

Range

192.168.8.0 - 192.168.8.63

Network Address

192.168.8.0

Broadcast Address

192.168.8.63

Hosts

192.168.8.1 – 192.168.8.62

Tuyến mặc định

Như chúng ta đã biết, các máy nằm trong cùng một mạng/mạng logic có thể giao tiếp trực tiếp với nhau thông qua giao thức IP.

Nhưng hãy cùng xem xét ví dụ dưới đây:

Mạng 1

192.168.10.0/24

Mạng 2

192.168.200.0/24

Trong trường hợp này, máy 192.168.10.20 không thể gửi trực tiếp gói đến 192.168.200.100 vì chúng nằm trên các mạng logic khác nhau.

Để kích hoạt giao tiếp này, bộ định tuyến (hoặc một tập hợp các bộ định tuyến) sẽ được sử dụng. Bộ định tuyến trong cấu hình này cũng có thể được gọi là cổng vì nó cung cấp một cổng giữa hai mạng. Thiết bị này có quyền truy cập vào cả hai mạng vì nó được định cấu hình với IP từ cả hai mạng (ví dụ như 192.168.10.1 và 192.168.200.1) và cũng vì chính lý do đó mà nó đóng vai trò trung gian trong giao tiếp này.

Để kích hoạt tính năng này, mỗi máy chủ trên mạng phải định cấu hình *tuyến mặc định*. Tuyến mặc định sẽ cho biết IP mà tất cả các gói có đích đến là IP không thuộc mạng logic của máy chủ phải được gửi tới.

Trong ví dụ trên, tuyến mặc định cho các máy trên mạng 192.168.10.0/24 sẽ là IP 192.168.10.1 - tức IP bộ định tuyến/cổng, trong khi tuyến mặc định cho các máy trên mạng 192.168.200.0/24 sẽ là 192.168.200.1.

Tuyến mặc định cũng được sử dụng để các máy trên mạng riêng tư (LAN) có thể truy cập Internet (WAN) thông qua bộ định tuyến.

Bài tập Hướng dẫn

1. Bằng cách sử dụng IP 172.16.30.230 và mặt nạ mạng 255.255.255.224, hãy xác định:

Ký hiệu CIDR cho mặt nạ mạng	
Địa chỉ mạng	
Địa chỉ truyền phát	
Số lượng IP có thể được sử dụng cho máy chủ trong mạng con này	

2. Cài đặt nào là bắt buộc trên máy chủ để cho phép một giao tiếp IP với một máy chủ trong một mạng logic khác?

Bài tập Mở rộng

1. Tại sao các dải IP bắt đầu bằng 127 và dải sau 224 không được bao gồm trong các hạng địa chỉ IP A, B hoặc C?

2. Một trong những trường rất quan trọng thuộc một gói IP là TTL (*Time To Live*). Chức năng của trường này là gì và nó hoạt động như thế nào?

3. Hãy giải thích chức năng của NAT và thời điểm nào nó sẽ được sử dụng.

Tóm tắt

Bài học này đã trình bày các khái niệm chính về giao thức IPv4 chịu trách nhiệm cho phép giao tiếp giữa các máy chủ trong cùng một mạng.

Các thao tác chính mà một chuyên gia phải biết để chuyển đổi IP ở các định dạng khác nhau cũng như có thể phân tích và thực hiện các cấu hình logic trên mạng và mạng con cũng đã được đề cập tới.

Các chủ đề sau đây đã được giải quyết:

- Hạng địa chỉ IP
- IP công cộng và riêng tư
- Cách chuyển đổi IP từ dạng thập phân sang dạng nhị phân và ngược lại
- Mặt nạ mạng (netmask)
- Cách xác định địa chỉ mạng và địa chỉ quảng bá từ IP và mặt nạ mạng

Tuyển mặc định

Đáp án Bài tập Hướng dẫn

1. Bằng cách sử dụng IP 172.16.30.230 và mặt nạ mạng 255.255.255.224, hãy xác định:

Ký hiệu CIDR cho mặt nạ mạng	27
Địa chỉ mạng	172.16.30.224
Địa chỉ truyền phát	172.16.30.255
Số lượng IP có thể được sử dụng cho máy chủ trong mạng con này	30

2. Cài đặt nào là bắt buộc trên máy chủ để cho phép một giao tiếp IP với một máy chủ trong một mạng logic khác?

Tuyến mặc định.

Đáp án Bài tập Mở rộng

1. Tại sao các dải IP bắt đầu bằng 127 và dải sau 224 không được bao gồm trong các hạng địa chỉ IP A, B hoặc C?

Phạm vi bắt đầu bằng 127 được dành riêng cho các địa chỉ lặp vòng được sử dụng để kiểm tra và liên lạc nội bộ giữa các tiến trình (chẳng hạn như địa chỉ 127.0.0.1). Ngoài ra, các địa chỉ vượt quá 224 cũng không được sử dụng làm địa chỉ máy chủ mà cho mục đích truyền đa hướng và các mục đích khác.

2. Một trong những trường rất quan trọng thuộc một gói IP là TTL (*Time To Live*). Chức năng của trường này là gì và nó hoạt động như thế nào?

TTL sẽ xác định thời gian tồn tại của một gói tin. Điều này được thực hiện thông qua một bộ đếm mà trong đó, giá trị ban đầu được xác định tại nguồn sẽ được giảm dần trong mỗi cổng/bộ định tuyến mà gói đi qua. Giá trị này còn được gọi là một “bước nhảy ngắn” (hop). Nếu bộ đếm này đạt tới giá trị 0 thì gói tin sẽ bị loại bỏ.

3. Hãy giải thích chức năng của NAT và thời điểm nào nó sẽ được sử dụng.

Tính năng NAT (*Dịch địa chỉ mạng*) cho phép các máy chủ trên mạng nội bộ sử dụng IP riêng tư truy cập vào Internet như thể chúng được kết nối trực tiếp với mạng đó (với IP công cộng được sử dụng trên cổng).



109.1 Bài 2

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	109 Những yếu tố cơ bản về Mạng
Mục tiêu:	109.1 Những yếu tố cơ bản của các Giao thức Internet
Bài:	2 trên 2

Giới thiệu

Ở phần mở đầu của chủ đề phụ này, chúng ta đã thấy rằng chồng giao thức TCP/IP có bao gồm một loạt các giao thức khác nhau. Cho đến nay, chúng ta đã nghiên cứu về giao thức IP cho phép giao tiếp giữa các máy thông qua địa chỉ IP, mặt nạ, tuyến, v.v.

Để một máy chủ có thể truy cập dịch vụ có sẵn trên một máy chủ khác, ngoài giao thức địa chỉ IP ở lớp mạng, nó sẽ cần phải sử dụng một giao thức ở lớp vận chuyển như giao thức TCP và UDP.

Các giao thức này sẽ thực hiện giao tiếp thông qua các cổng mạng. Vì vậy, ngoài việc xác định IP nguồn và IP đích, các cổng nguồn và đích cũng sẽ được sử dụng để truy cập một dịch vụ.

Cổng sẽ được xác định bằng một trường 16 bit, từ đó cung cấp giới hạn 65.535 cổng có thể có. Các dịch vụ (đích) sử dụng cổng từ 1 đến 1023 được gọi là *cổng đặc quyền* vì chúng có quyền truy cập gốc vào hệ thống. Nguồn gốc của kết nối sẽ sử dụng phạm vi cổng từ 1024 đến 65.535 được gọi là *cổng không có đặc quyền* hoặc *cổng ổ nối*.

Các cổng được sử dụng bởi từng loại dịch vụ sẽ được chuẩn hóa và kiểm soát bởi IANA (*Cơ quan cung cấp số hiệu Internet*). Điều này có nghĩa là trên bất kỳ một hệ thống nào, cổng 22 cũng sẽ

được sử dụng bởi dịch vụ SSH, cổng 80 sẽ được sử dụng bởi dịch vụ HTTP, v.v.

Bảng dưới đây có chứa các dịch vụ chính và cổng tương ứng của chúng.

Cổng	Dịch vụ
20	FTP (dữ liệu)
21	FTP (điều khiển)
22	SSH (Vỏ ổ nối an toàn)
23	Telnet (Kết nối từ xa không cần mã hóa)
25	SMTP (Giao thức truyền tải thư đơn giản), Sending Mails (Gửi Thư)
53	DNS (Hệ thống tên miền)
80	HTTP (Giao thức truyền tải siêu văn bản)
110	POP3 (Giao thức bưu điện), Receiving Mails (Nhận Thư)
123	NTP (Giao thức thời gian mạng)
139	Netbios
143	IMAP (Giao thức truy cập tin nhắn Internet), Accessing Mails (Truy cập Thư)
161	SNMP (Giao thức quản lý mạng đơn giản)
162	Thông báo SNMPTRAP, SNMP
389	LDAP (Giao thức truy cập cấu trúc thư mục)
443	HTTPS (HTTP an toàn)
465	SMTPS (SMTP an toàn)
514	RSH (Vỏ từ xa)
636	LDAPS (LDAP an toàn)
993	IMAPS (IMAP an toàn)
995	POP3S (POP3 an toàn)

Trên một hệ thống Linux, các cổng dịch vụ tiêu chuẩn sẽ được liệt kê trong tệp `/etc/services`.

Việc xác định cổng đích mong muốn trong kết nối có thể được thực hiện bằng ký tự `:` (dấu hai chấm) sau địa chỉ IPv4. Do đó, khi tìm kiếm quyền truy cập vào dịch vụ HTTPS được cung cấp bởi

máy chủ IP 200.216.10.15, máy khách sẽ phải gửi yêu cầu đến đích 200.216.10.15:443.

Các dịch vụ được liệt kê ở trên và tất cả các dịch vụ khác đều sử dụng giao thức truyền tải theo các đặc điểm mà dịch vụ yêu cầu, trong đó TCP và UDP là các dịch vụ chính.

Giao thức Điều khiển Truyền nhận (TCP)

TCP là một giao thức truyền tải định hướng kết nối. Điều này có nghĩa là một kết nối sẽ được thiết lập giữa máy khách thông qua cổng ổ nối và dịch vụ thông qua cổng tiêu chuẩn dịch vụ. Giao thức này có nhiệm vụ đảm bảo rằng tất cả các gói sẽ được gửi đúng cách, xác minh tính toàn vẹn và thứ tự của các gói bao gồm cả việc truyền lại các gói bị mất do lỗi mạng.

Do đó, ứng dụng sẽ không cần thực hiện kiểm soát luồng dữ liệu này vì nó đã được đảm bảo bởi giao thức TCP.

Giao thức Gói dữ liệu Người dùng (UDP)

UDP thiết lập một kết nối giữa máy khách và dịch vụ nhưng không kiểm soát việc truyền dữ liệu của kết nối đó. Nói cách khác, nó sẽ không kiểm tra xem các gói có bị thất lạc hay có được truyền theo đúng thứ tự hay không, v.v. Ứng dụng sẽ phải chịu trách nhiệm thực hiện các biện pháp kiểm soát cần thiết.

Vì có ít quyền kiểm soát hơn nên UDP cho phép luồng dữ liệu có hiệu suất tốt hơn. Điều này rất quan trọng đối với một số loại dịch vụ.

Giao thức Thông điệp Điều khiển Internet (ICMP)

ICMP là một giao thức lớp mạng trong chồng TCP/IP với chức năng chính là phân tích và kiểm soát các thành phần mạng. Ví dụ:

- Kiểm soát lưu lượng tải
- Phát hiện các đích đến không thể truy cập được
- Chuyển hướng tuyến
- Kiểm tra trạng thái của máy chủ từ xa

Đây là giao thức được sử dụng bởi lệnh `ping` và sẽ được thảo luận tới trong một chủ đề phụ khác.

IPv6

Cho đến nay, chúng ta đã nghiên cứu về phiên bản thứ 4 của giao thức IP - tức IPv4. Đây là phiên

bản tiêu chuẩn được sử dụng trong mọi môi trường mạng và Internet. Tuy nhiên, nó cũng có những hạn chế, đặc biệt là về số lượng địa chỉ khả dụng; với thực tế hiện tại khi mà tất cả các thiết bị đều sẽ được kết nối với Internet bằng một cách nào đó (xem IoT), việc sử dụng phiên bản thứ 6 của giao thức IP - hay IPv6 - đang ngày càng trở nên phổ biến.

IPv6 đang mang đến một loạt các thay đổi, các cách triển khai và các tính năng mới cũng như một cách thể hiện địa chỉ mới.

Mỗi một địa chỉ IPv6 sẽ có 128 bit được chia thành 8 nhóm 16 bit và được biểu thị bằng giá trị thập lục phân.

Ví dụ:

```
2001:0db8:85a3:08d3:1319:8a2e:0370:7344
```

Cụm viết tắt

IPv6 đã xác định các cách để rút ngắn địa chỉ trong một số trường hợp. Hãy cùng xem lại địa chỉ sau:

```
2001:0db8:85a3:0000:0000:0000:0000:7344
```

Khả năng đầu tiên là giảm chuỗi từ `0000` xuống chỉ còn `0`; từ đây ta có:

```
2001:0db8:85a3:0:0:0:0:7344
```

Ngoài ra, trong trường hợp chuỗi nhóm có giá trị `0`, chúng có thể được bỏ qua như sau:

```
2001:0db8:85a3::7344
```

Tuy nhiên, cụm viết tắt cuối cùng này chỉ có thể được thực hiện một lần trong địa chỉ. Hãy xem ví dụ:

```
2001:0db8:85a3:0000:0000:1319:0000:7344
```

```
2001:0db8:85a3:0:0:1319:0:7344
```

```
2001:0db8:85a3::1319:0:7344
```

Các loại địa chỉ IPv6

IPv6 phân loại địa chỉ thành 3 loại:

Truyền Đơn hướng

Xác định một giao diện mạng duy nhất. Theo mặc định, 64 bit ở bên trái sẽ xác định mạng và 64 bit ở bên phải sẽ xác định giao diện.

Truyền Đa hướng

Xác định một tập hợp các giao diện mạng. Một gói được gửi đến địa chỉ đa hướng sẽ được gửi đến tất cả các giao diện thuộc nhóm đó. Mặc dù cũng tương tự như địa chỉ truyền phát nhưng chúng ta không được nhầm lẫn giữa hai loại bởi địa chỉ truyền phát không tồn tại trong giao thức IPv6.

Truyền chọn lọc

Địa chỉ này cũng xác định một tập hợp các giao diện trên mạng nhưng gói được chuyển tiếp đến một địa chỉ *truyền chọn lọc* sẽ chỉ được gửi đến một địa chỉ trong tập hợp đó chứ không phải cho tất cả mọi người.

Sự khác biệt giữa IPv4 và IPv6

Ngoài địa chỉ, một số khác biệt khác cũng có thể được chỉ ra giữa phiên bản thứ 4 và thứ 6 của IP. Dưới đây là một số trong số chúng:

- Các cổng dịch vụ đều tuân theo cùng một tập hợp các tiêu chuẩn và giao thức (TCP, UDP), sự khác biệt chỉ nằm ở cách thể hiện IP và bộ cổng. Trong IPv6, địa chỉ IP phải được nằm trong [] (dấu ngoặc vuông):

IPv4

```
200.216.10.15:443
```

IPv6

```
[2001:0db8:85a3:08d3:1319:8a2e:0370:7344]:443
```

- IPv6 không triển khai tính năng truyền phát chính xác như trong IPv4. Tuy nhiên, kết quả tương tự vẫn có thể đạt được bằng cách gửi gói đến địa chỉ `ff02::1` và từ đó gửi đến tất cả các máy chủ trên mạng cục bộ. Điều này cũng tương tự như việc sử dụng `224.0.0.1` trên IPv4 làm điểm đích để truyền phát đa hướng.
- Thông qua tính năng SLAAC (*Tự động cấu hình địa chỉ không trạng thái*), máy chủ IPv6 có thể tự định cấu hình.

- Trường TTL (*Time to Live*) của IPv4 đã được thay thế bằng “Hop Limit” trong tiêu đề của IPv6.
- Tất cả các giao diện IPv6 đều có một địa chỉ cục bộ được gọi là địa chỉ liên kết cục bộ với tiền tố `fe80::/10`.
- IPv6 triển khai *Giao thức Khám phá Vùng lân cận* (NDP) cũng tương tự như ARP được IPv4 sử dụng nhưng có nhiều chức năng hơn.

Bài tập Hướng dẫn

1. Cổng nào là mặc định cho giao thức SMTP?

2. Có bao nhiêu cổng khác nhau có sẵn trong một hệ thống?

3. Giao thức truyền tải nào sẽ đảm bảo rằng tất cả các gói được phân phối đúng cách và xác minh được tính toàn vẹn và thứ tự của các gói?

4. Loại địa chỉ IPv6 nào được sử dụng để gửi gói đến tất cả các giao diện thuộc nhóm máy chủ?

Bài tập Mở rộng

1. Hãy nêu 4 ví dụ về các dịch vụ sử dụng giao thức TCP theo mặc định.

2. Tên của trường trên gói tiêu đề IPv6 triển khai cùng một tài nguyên TTL trên IPv4 là gì?

--

3. Giao thức Khám phá Vùng lân cận (NDP) có thể tìm thấy loại thông tin nào?

--

Tóm tắt

Bài học này đã trình bày về các giao thức và dịch vụ truyền tải chính được sử dụng trên chồng TCP/IP.

Một chủ đề quan trọng khác là phiên bản thứ 6 của Giao thức IP bao gồm địa chỉ IPv6 và những điểm khác biệt chính so với IPv4.

Các chủ đề sau đây đã được giải quyết:

- Mối tương quan giữa số Cổng và Dịch vụ
- TCP (Giao thức Điều khiển Truyền nhận)
- UDP (Giao thức Gói Dữ liệu Người dùng)
- ICMP (Giao thức Thông điệp Điều khiển Internet)
- Địa chỉ IPv6 và cụm viết tắt của nó
- Các loại địa chỉ IPv6
- Sự khác biệt chính giữa IPv4 và IPv6

Đáp án Bài tập Hướng dẫn

1. Cổng nào là mặc định cho giao thức SMTP?

25.

2. Có bao nhiêu cổng khác nhau có sẵn trong một hệ thống?

65535.

3. Giao thức truyền tải nào sẽ đảm bảo rằng tất cả các gói được phân phối đúng cách và xác minh được tính toàn vẹn và thứ tự của các gói?

TCP.

4. Loại địa chỉ IPv6 nào được sử dụng để gửi gói đến tất cả các giao diện thuộc nhóm máy chủ?

Địa chỉ truyền đa hướng.

Đáp án Bài tập Mở rộng

1. Hãy nêu 4 ví dụ về các dịch vụ sử dụng giao thức TCP theo mặc định.

FTP, SMTP, HTTP, POP3, IMAP, SSH

2. Tên của trường trên gói tiêu đề IPv6 triển khai cùng một tài nguyên TTL trên IPv4 là gì?

Hop Limit.

3. Giao thức Khám phá Vùng lân cận (NDP) có thể tìm thấy loại thông tin nào?

NDP có thể lấy nhiều loại thông tin khác nhau từ mạng bao gồm các nút khác, địa chỉ trùng lặp, tuyến, máy chủ DNS, cổng, v.v.



109.2 Cấu hình Mạng liên tục

Tham khảo các mục tiêu LPI

[LPIC-1 version 5.0, Exam 102, Objective 109.2](#)

Khối lượng

4

Các lĩnh vực kiến thức chính

- Hiểu cấu hình máy chủ TCP/IP cơ bản
- Định cấu hình cấu hình mạng ethernet và wi-fi bằng NetworkManager
- Nhận thức về systemd-networkd

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- `/etc/hostname`
- `/etc/hosts`
- `/etc/nsswitch.conf`
- `/etc/resolv.conf`
- `nmcli`
- `hostnamectl`
- `ifup`
- `ifdown`



109.2 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	109 Những yếu tố cơ bản về Mạng
Mục tiêu:	109.2 Cấu hình Mạng liên tục
Bài:	1 trên 2

Giới thiệu

Trong bất kỳ một mạng TCP/IP nào, mọi nút đều phải định cấu hình bộ điều hợp mạng của nó để phù hợp với yêu cầu của mạng; nếu không, chúng sẽ không thể giao tiếp với nhau. Do đó, quản trị viên hệ thống phải cung cấp cấu hình cơ bản để hệ điều hành có thể thiết lập giao diện mạng phù hợp cũng như nhận dạng chính nó và các tính năng cơ bản của mạng mỗi khi khởi động.

Cài đặt mạng không liên quan đến hệ điều hành nhưng hệ điều hành có phương pháp riêng của nó để lưu trữ và áp dụng các cài đặt này. Các hệ thống Linux dựa vào các cấu hình được lưu trữ trong các tệp văn bản thuần túy ở thư mục `/etc` để hiển thị kết nối mạng trong thời gian khởi động. Điều chúng ta cần biết là các tệp này được sử dụng như thế nào để tránh mất kết nối do sai cấu hình cục bộ.

Giao diện Mạng

Giao diện mạng là thuật ngữ mà hệ điều hành dùng để chỉ kênh giao tiếp được định cấu hình để hoạt động với phần cứng mạng được gắn vào hệ thống (chẳng hạn như thiết bị ethernet hoặc wi-fi). Ngoại lệ cho điều này là giao diện *lặp vòng* (loopback) - một giao diện mà hệ điều hành sẽ sử dụng khi cần thiết lập kết nối với chính nó. Mục đích chính của giao diện mạng là cung cấp một

tuyến đường để qua đó, dữ liệu cục bộ có thể được gửi cũng như có thể nhận được dữ liệu từ xa. Nếu giao diện mạng được cấu hình sai cách, hệ điều hành sẽ không thể giao tiếp với các máy khác trong mạng.

Trong hầu hết các trường hợp, cài đặt giao diện chính xác sẽ được xác định theo mặc định hoặc được tùy chỉnh trong quá trình cài đặt hệ điều hành. Tuy nhiên, những cài đặt này thường cần được kiểm tra hoặc thậm chí là sửa đổi khi quá trình giao tiếp hoạt động không bình thường hoặc khi hoạt động của giao diện cần được tùy chỉnh.

Có nhiều lệnh Linux để liệt kê các giao diện mạng nào hiện có trên hệ thống, nhưng không phải tất cả các lệnh này đều có sẵn trong mọi bản phân phối. Tuy nhiên, lệnh `ip` là một phần của bộ công cụ mạng cơ bản đi kèm với tất cả các bản phân phối Linux và có thể được sử dụng để liệt kê các giao diện mạng. Lệnh hoàn chỉnh để hiển thị các giao diện là `ip link show`:

```
$ ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp3s5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT
group default qlen 1000
    link/ether 00:16:3e:8d:2b:5b brd ff:ff:ff:ff:ff:ff
```

Nếu có, lệnh `nmcli device` cũng có thể được sử dụng:

```
$ nmcli device
DEVICE      TYPE      STATE      CONNECTION
enp3s5      ethernet  connected  Gigabit Powerline Adapter
lo          loopback  unmanaged  --
```

Các lệnh hiển thị trong các ví dụ không sửa đổi bất kỳ cài đặt nào trong hệ thống; do đó, chúng có thể được thực thi bởi người dùng không có đặc quyền. Cả hai lệnh đều liệt kê hai giao diện mạng: `lo` (giao diện lặp vòng) và `enp3s5` (giao diện ethernet).

Máy tính để bàn và máy tính xách tay chạy Linux thường có hai hoặc ba giao diện mạng được xác định trước: một giao diện cho giao diện ảo lặp vòng và các giao diện còn lại được gán cho phần cứng mạng được hệ thống tìm thấy. Mặt khác, máy chủ và thiết bị mạng chạy Linux có thể có hàng chục giao diện mạng, nhưng các nguyên tắc giống nhau đều được áp dụng cho tất cả các giao diện này. Tính trừu tượng do hệ điều hành cung cấp cho phép thiết lập giao diện mạng thông qua các phương pháp giống nhau bất kể phần cứng cơ bản là gì.

Tuy nhiên, việc nắm vững một cách chi tiết về phần cứng cơ bản của giao diện có thể sẽ hữu ích

trong việc hiểu rõ hơn điều gì đang xảy ra khi giao tiếp không hoạt động được như mong đợi. Chẳng hạn như trong một hệ thống có sẵn nhiều giao diện mạng, chúng ta không thể biết rõ giao diện nào tương ứng với wi-fi và giao diện nào tương ứng với ethernet. Vì lý do này, Linux sử dụng quy ước đặt tên giao diện để giúp xác định giao diện mạng nào tương ứng với thiết bị và cổng nào.

Tên Giao diện

Các bản phân phối Linux cũ hơn có tên giao diện mạng ethernet là `eth0`, `eth1`, v.v., được đánh số theo thứ tự mà hạt nhân xác định các thiết bị. Các giao diện không dây được đặt tên là `wlan0`, `wlan1`, v.v. Tuy nhiên, quy ước đặt tên này không làm rõ được một số vấn đề như cổng ethernet cụ thể nào sẽ phù hợp với giao diện `eth0`. Tùy thuộc vào cách phát hiện phần cứng, thậm chí hai giao diện mạng cũng có thể đổi tên cho nhau sau khi khởi động lại.

Để khắc phục sự mơ hồ này, các hệ thống Linux gần đây đã sử dụng quy ước đặt tên có thể dự đoán được cho các giao diện mạng và tạo nên một mối quan hệ chặt chẽ hơn giữa tên giao diện và kết nối phần cứng cơ bản.

Trong các bản phân phối Linux sử dụng cơ chế đặt tên `systemd`, tất cả các tên giao diện đều bắt đầu bằng tiền tố gồm hai ký tự biểu thị loại giao diện:

en

Ethernet

ib

InfiniBand

sl

Giao thức Internet đường dây nối tiếp (slip)

wl

Mạng không dây cục bộ (WLAN)

ww

Mạng không dây diện rộng (WWAN)

Từ mức độ ưu tiên cao hơn đến thấp hơn, hệ điều hành sử dụng các quy tắc sau để đặt tên và đánh số các giao diện mạng:

1. Đặt tên giao diện theo chỉ mục do BIOS hoặc chương trình cơ sở của thiết bị nhúng cung cấp (ví dụ: `eno1`).
2. Đặt tên giao diện theo chỉ mục khe cắm PCI express như được cung cấp bởi BIOS hoặc chương

- trình cơ sở (ví dụ: ens1).
- Đặt tên giao diện theo địa chỉ của nó tại cổng bus tương ứng (ví dụ: enp3s5).
 - Đặt tên giao diện theo địa chỉ MAC của giao diện (ví dụ: enx78e7d1ea46da).
 - Đặt tên cho giao diện bằng cách sử dụng quy ước cũ (ví dụ: eth0).

Chúng ta hoàn toàn có thể giả định rằng giao diện mạng enp3s5 được đặt tên như vậy vì nó không phù hợp với hai phương thức đặt tên đầu tiên; do đó, địa chỉ của nó trong cổng bus và khe cắm tương ứng đã được sử dụng thay thế. Địa chỉ thiết bị 03:05.0 được tìm thấy trong đầu ra của lệnh `lspci` đã cho biết thiết bị liên kết:

```
$ lspci | fgrep Ethernet
03:05.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL-8110SC/8169SC Gigabit Ethernet (rev 10)
```

Giao diện mạng được tạo bởi chính nhân Linux nhưng có nhiều lệnh có thể được sử dụng để tương tác với chúng. Thông thường, quá trình cấu hình sẽ diễn ra tự động và không cần thay đổi cài đặt theo cách thủ công. Tuy nhiên, với tên của giao diện, chúng ta có thể cho hạt nhân biết cách tiến hành cấu hình nó nếu cần.

Quản lý Giao diện

Qua nhiều năm, một số chương trình đã được phát triển để tương tác với các tính năng mạng do nhân Linux cung cấp. Lệnh `ifconfig` cũ vẫn có thể được sử dụng để thực hiện các truy vấn và cấu hình giao diện đơn giản nhưng hiện tại nó không còn được sử dụng nữa do khả năng hỗ trợ hạn chế đối với các giao diện không phải ethernet. Lệnh `ifconfig` được thay thế bằng lệnh `ip` có khả năng quản lý nhiều khía cạnh khác của giao diện TCP/IP như các tuyến và đường hầm.

Các khả năng đa dạng của lệnh `ip` có thể là vượt quá mức cần thiết đối với hầu hết các tác vụ thông thường. Do đó, chúng ta còn có các lệnh phụ trợ để tạo điều kiện thuận lợi cho việc kích hoạt và cấu hình các giao diện mạng. Các lệnh `ifup` và `ifdown` có thể được sử dụng để định cấu hình giao diện mạng dựa trên các định nghĩa giao diện được tìm thấy trong tệp `/etc/network/interfaces`. Mặc dù cũng có thể được gọi theo cách thủ công nhưng các lệnh này thường được thực thi tự động trong quá trình khởi động hệ thống.

Tất cả các giao diện mạng được quản lý bởi `ifup` và `ifdown` phải được liệt kê trong tệp `/etc/network/interfaces`. Định dạng được sử dụng trong tệp rất đơn giản: các dòng bắt đầu bằng từ `auto` được sử dụng để xác định các giao diện vật lý được hiển thị khi `ifup` được thực thi với tùy chọn `-a`. Tên giao diện phải theo sau từ `auto` trên cùng một dòng. Tất cả các giao diện được đánh dấu `auto` đều sẽ được hiển thị khi khởi động theo thứ tự được liệt kê.

WARNING

Các phương thức cấu hình mạng được `ifup` và `ifdown` sử dụng không được chuẩn hóa trong tất cả các bản phân phối Linux. Ví dụ: CentOS sẽ giữ các cài đặt giao diện trong các tệp riêng lẻ trong thư mục `/etc/sysconfig/network-scripts/` và định dạng cấu hình được sử dụng trong chúng sẽ hơi khác so với định dạng được sử dụng trong `/etc/network/interfaces`.

Cấu hình giao diện thực tế được viết bằng một dòng khác bắt đầu bằng từ `iface`, theo sau là tên giao diện, tên của họ địa chỉ mà giao diện sử dụng và tên của phương thức được sử dụng để định cấu hình giao diện. Ví dụ sau đây cho thấy một tệp cấu hình cơ bản cho giao diện `lo` (loopback) và `enp3s5`:

```
auto lo
iface lo inet loopback

auto enp3s5
iface enp3s5 inet dhcp
```

Họ địa chỉ phải là `inet` cho mạng TCP/IP nhưng mạng IPX (`ipx`) và mạng IPv6 (`inet6`) cũng được hỗ trợ. Giao diện lặp vòng sử dụng phương thức cấu hình `loopback`. Với phương thức `dhcp`, giao diện sẽ sử dụng cài đặt IP do máy chủ DHCP của mạng cung cấp. Các cài đặt từ cấu hình ví dụ sẽ cho phép thực thi lệnh `ifup` bằng cách sử dụng tên giao diện `enp3s5` làm đối số:

```
# ifup enp3s5
Internet Systems Consortium DHCP Client 4.4.1
Copyright 2004-2018 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/enp3s5/00:16:3e:8d:2b:5b
Sending on   LPF/enp3s5/00:16:3e:8d:2b:5b
Sending on   Socket/fallback
DHCPDISCOVER on enp3s5 to 255.255.255.255 port 67 interval 4
DHCPOFFER of 10.90.170.158 from 10.90.170.1
DHCPREQUEST for 10.90.170.158 on enp3s5 to 255.255.255.255 port 67
DHCPACK of 10.90.170.158 from 10.90.170.1
bound to 10.90.170.158 -- renewal in 1616 seconds.
```

Trong ví dụ này, phương thức được chọn cho giao diện `enp3s5` là `dhcp`; vì vậy, lệnh `ifup` đã gọi chương trình máy khách DHCP để lấy cài đặt IP từ máy chủ DHCP. Tương tự, lệnh `ifdown enp3s5` có thể được sử dụng để tắt giao diện.

Trong các mạng không có máy chủ DHCP, phương pháp `static` có thể được sử dụng thay thế và cài đặt IP được cung cấp thủ công trong `/etc/network/interfaces`. Ví dụ:

```
iface enp3s5 inet static
    address 192.168.1.2/24
    gateway 192.168.1.1
```

Các giao diện sử dụng phương thức `static` không cần lệnh `auto` tương ứng vì chúng sẽ được hiển thị bất cứ khi nào phát hiện thấy phần cứng mạng.

Nếu cùng một giao diện có nhiều mục nhập `iface` thì tất cả các địa chỉ và tùy chọn đã định cấu hình sẽ được áp dụng khi hiển thị giao diện đó. Điều này rất hữu ích trong việc định cấu hình cả địa chỉ IPv4 và IPv6 trên cùng một giao diện cũng như định cấu hình nhiều địa chỉ cùng loại trên một giao diện.

Tên cục bộ và Tên kết nối từ xa

Một thiết lập TCP/IP khả dụng chỉ là bước đầu tiên hướng tới khả năng sử dụng toàn bộ mạng. Ngoài việc có thể xác định các nút trên mạng bằng số IP của chúng, hệ thống phải có khả năng xác định chúng bằng những tên mà con người có thể hiểu một cách dễ dàng hơn.

Tên mà hệ thống tự nhận dạng có thể tùy chỉnh được và chúng ta nên xác định tên đó ngay cả khi máy không có ý định tham gia mạng. Tên cục bộ thường trùng với tên mạng của máy nhưng điều này không nhất thiết lúc nào cũng đúng. Nếu tệp `/etc/hostname` tồn tại, hệ điều hành sẽ sử dụng nội dung của dòng đầu tiên làm tên cục bộ của nó, sau đó nó sẽ được gọi đơn giản là *hostname* (tên máy chủ). Các dòng bắt đầu bằng `#` bên trong `/etc/hostname` sẽ bị bỏ qua.

Tệp `/etc/hostname` có thể được chỉnh sửa trực tiếp nhưng tên máy chủ của máy cũng có thể được xác định bằng lệnh `hostnamectl`. Khi được cung cấp lệnh phụ `set-hostname`, lệnh `hostnamectl` sẽ lấy tên được cung cấp làm đối số và ghi nó vào `/etc/hostname`:

```
# hostnamectl set-hostname storage
# cat /etc/hostname
storage
```

Tên máy chủ được xác định trong `/etc/hostname` là tên máy chủ *tĩnh* - tức là tên được sử dụng để khởi tạo tên máy chủ của hệ thống khi khởi động. Tên máy chủ tĩnh có thể là một chuỗi dạng tự do có độ dài tối đa 64 ký tự. Tuy nhiên, khuyến nghị dành cho người dùng là nó chỉ nên bao gồm các ký tự chữ thường ASCII và không có dấu cách hoặc dấu chấm. Nó cũng nên giới hạn ở định dạng được phép cho nhãn tên miền DNS (mặc dù đây không phải là một yêu cầu nghiêm ngặt).

Lệnh `hostnamectl` có thể đặt hai loại tên máy chủ khác ngoài tên máy chủ tĩnh:

Tên máy chủ thân thiện (pretty)

Không giống như tên máy chủ tĩnh, tên máy chủ thân thiện có thể bao gồm tất cả các loại ký tự đặc biệt. Nó có thể được sử dụng để đặt tên mô tả nhiều hơn cho máy, ví dụ như “LAN Shared Storage”:

```
# hostnamectl --pretty set-hostname "LAN Shared Storage"
```

Tên máy chủ tạm thời (transient)

Được sử dụng khi tên máy chủ tĩnh không được đặt hoặc được để mặc định là `localhost`. Tên máy chủ tạm thời thường là tên được đặt cùng với các cấu hình tự động khác, nhưng nó cũng có thể được sửa đổi bằng lệnh `hostnamectl`. Ví dụ:

```
# hostnamectl --transient set-hostname generic-host
```

Nếu cả tùy chọn `--pretty` hay `--transient` đều không được sử dụng thì cả ba loại tên máy chủ sẽ được đặt thành tên đã cho. Để đặt tên máy chủ tĩnh (chứ không phải tên thân thiện và tạm thời), chúng ta nên sử dụng tùy chọn `--static`. Trong mọi trường hợp, chỉ tên máy chủ tĩnh mới được lưu trữ trong tệp `/etc/hostname`. Lệnh `hostnamectl` cũng có thể được sử dụng để hiển thị các thông tin mô tả và nhận dạng khác nhau về hệ thống đang chạy:

```
$ hostnamectl status
  Static hostname: storage
  Pretty hostname: LAN Shared Storage
  Transient hostname: generic-host
    Icon name: computer-server
     Chassis: server
  Machine ID: d91962a957f749bbaf16da3c9c86e093
   Boot ID: 8c11dcab9c3d4f5aa53f4f4e8fdc6318
  Operating System: Debian GNU/Linux 10 (buster)
     Kernel: Linux 4.19.0-8-amd64
  Architecture: x86-64
```

Đây là hành động mặc định của lệnh `hostnamectl` nên lệnh phụ `status` có thể được bỏ qua.

Về tên của các nút mạng từ xa, có hai cách cơ bản mà hệ điều hành có thể thực hiện để khớp tên và số IP: sử dụng nguồn cục bộ hoặc sử dụng máy chủ từ xa để dịch tên thành số IP và ngược lại. Các phương thức có thể bổ sung cho nhau và thứ tự ưu tiên của chúng sẽ được xác định trong tệp

cấu hình *Name Service Switch* (Chuyển đổi dịch vụ tên): `/etc/nsswitch.conf`. Tập này sẽ được hệ thống và ứng dụng sử dụng để xác định không chỉ các nguồn trùng khớp tên IP mà còn cả các nguồn để lấy thông tin dịch vụ tên trong một loạt danh mục được gọi là *cơ sở dữ liệu*.

Cơ sở dữ liệu *các máy chủ* sẽ theo dõi ánh xạ giữa tên máy chủ và số hiệu máy chủ. Dòng bên trong `/etc/nsswitch.conf` bắt đầu bằng `hosts` sẽ xác định các dịch vụ chịu trách nhiệm cung cấp các liên kết cho nó:

```
hosts: files dns
```

Trong mục ví dụ này, `files` và `dns` là tên dịch vụ chỉ định cách hoạt động của quá trình tra cứu tên máy chủ. Đầu tiên, hệ thống sẽ tìm kiếm các kết quả trùng khớp trong các tệp cục bộ; sau đó, nó sẽ yêu cầu dịch vụ DNS tìm các kết quả trùng khớp.

Tệp cục bộ dành cho cơ sở dữ liệu máy chủ là `/etc/hosts` - một tệp văn bản đơn giản liên kết địa chỉ IP với tên máy chủ, mỗi địa chỉ IP ở trên một dòng. Dưới đây là một ví dụ:

```
127.0.0.1 localhost
```

Số IP `127.0.0.1` là địa chỉ mặc định cho giao diện lặp vòng, do đó mà nó được liên kết với tên `localhost`.

Chúng ta cũng có thể liên kết các bí danh tùy chọn với cùng một IP. Bí danh cung cấp các cách viết khác hoặc tên máy chủ ngắn hơn và nên được thêm vào cuối dòng như sau:

```
192.168.1.10 foo.mydomain.org foo
```

Các quy tắc định dạng cho tệp `/etc/hosts` là:

- Các trường của mục nhập được phân tách bằng số lượng khoảng trống và/hoặc ký tự tab bất kỳ.
- Phần văn bản từ ký tự `#` cho đến cuối dòng là một chú thích và sẽ bị bỏ qua.
- Tên máy chủ chỉ có thể chứa các ký tự chữ, số, dấu trừ và dấu chấm.
- Tên máy chủ phải bắt đầu bằng một ký tự chữ cái và kết thúc bằng một ký tự chữ hoặc số.

Địa chỉ IPv6 cũng có thể được thêm vào `/etc/hosts`. Mục sau đây ám chỉ địa chỉ lặp vòng IPv6:

```
::1 localhost ip6-localhost ip6-loopback
```

Theo đặc tả dịch vụ `files`, đặc tả `dns` sẽ khiến hệ thống yêu cầu một dịch vụ DNS để có được liên kết tên/IP như mong muốn. Tập hợp các thủ tục chịu trách nhiệm cho phương thức này được gọi là *trình phân giải* (resolver) và tệp cấu hình của nó là `/etc/resolv.conf`. Ví dụ sau đây sẽ cho thấy một `/etc/resolv.conf` chung chứa các mục nhập cho máy chủ DNS công cộng của Google:

```
nameserver 8.8.4.4
nameserver 8.8.8.8
```

Như trong ví dụ, từ khóa `nameserver` cho biết địa chỉ IP của máy chủ DNS. Thực tế chỉ cần một máy chủ tên miền nhưng chúng ta cũng có thể có tối đa ba máy chủ. Những máy chủ tên miền bổ sung sẽ được sử dụng như một phương án dự phòng. Nếu không có mục nhập máy chủ tên miền nào, hành vi mặc định sẽ là sử dụng máy chủ tên miền trên máy cục bộ.

Trình phân giải có thể được cấu hình để tự động thêm miền vào tên trước khi tham khảo chúng trên máy chủ tên miền. Ví dụ:

```
nameserver 8.8.4.4
nameserver 8.8.8.8
domain mydomain.org
search mydomain.net mydomain.com
```

Mục nhập `domain` sẽ đặt `mydomain.org` làm tên miền cục bộ; vì vậy, các truy vấn tên trong miền này sẽ được phép sử dụng các tên ngắn liên quan đến miền cục bộ. Mục `search` cũng có mục đích tương tự nhưng nó sẽ chấp nhận một danh sách các miền để thử khi một tên ngắn được cung cấp. Theo mặc định, nó sẽ chỉ chứa tên miền cục bộ.

Bài tập Hướng dẫn

1. Những lệnh nào có thể được sử dụng để liệt kê các bộ điều hợp mạng có trong hệ thống?

2. Bộ điều hợp mạng với tên giao diện là `wlo1` thuộc loại gì?

3. Tập `/etc/network/interfaces` đóng vai trò gì trong quá trình khởi động?

4. Mục nào trong `/etc/network/interfaces` sẽ cấu hình giao diện `eno1` để nhận cài đặt IP của nó bằng DHCP?

Bài tập Mở rộng

1. Làm thế nào mà lệnh `hostnamectl` có thể được sử dụng để chỉ thay đổi tên máy chủ *tính* của máy cục bộ thành `firewall`?

2. Những chi tiết nào ngoài tên máy chủ có thể được sửa đổi bằng lệnh `hostnamectl`?

3. Mục nào trong `/etc/hosts` sẽ liên kết cả hai tên `firewall` và `router` với IP `10.8.0.1`?

4. Làm cách nào để có thể sửa đổi tệp `/etc/resolv.conf` để gửi tất cả các yêu cầu DNS đến `1.1.1.1`?

Tóm tắt

Bài học này đã trình bày cách thực hiện các thay đổi bền vững với cấu hình mạng cục bộ bằng cách sử dụng các tệp và lệnh Linux tiêu chuẩn. Linux luôn giả định các cài đặt TCP/IP sẽ ở những vị trí cụ thể và người dùng có thể sẽ cần thay đổi chúng khi cài đặt mặc định không phù hợp. Bài học đã đi qua các chủ đề sau:

- Cách Linux xác định các giao diện mạng.
- Kích hoạt giao diện trong quá trình khởi động và cấu hình IP cơ bản.
- Cách hệ điều hành liên kết tên với máy chủ.

Các khái niệm, lệnh và quy trình đã được giải quyết là:

- Quy ước đặt tên giao diện.
- Liệt kê các giao diện mạng với `ip` và `nmcli`.
- Kích hoạt giao diện với `ifup` và `ifdown`.
- Lệnh `hostnamectl` và tệp `/etc/hostname`.
- Các tệp `/etc/nsswitch.conf`, `/etc/hosts` và `/etc/resolv.conf`.

Đáp án Bài tập Hướng dẫn

1. Những lệnh nào có thể được sử dụng để liệt kê các bộ điều hợp mạng có trong hệ thống?

Các lệnh `ip link show`, `nmcli device` và `ifconfig` kế thừa.

2. Bộ điều hợp mạng với tên giao diện là `wlo1` thuộc loại gì?

Vì có tên được bắt đầu bằng `wl` nên nó là một bộ điều hợp mạng LAN không dây.

3. Tập `/etc/network/interfaces` đóng vai trò gì trong quá trình khởi động?

Nó có các cấu hình được sử dụng bởi lệnh `ifup` để kích hoạt các giao diện tương ứng trong thời gian khởi động.

4. Mục nào trong `/etc/network/interfaces` sẽ cấu hình giao diện `eno1` để nhận cài đặt IP của nó bằng DHCP?

Dòng `iface eno1 inet dhcp`.

Đáp án Bài tập Mở rộng

1. Làm thế nào mà lệnh `hostnamectl` có thể được sử dụng để chỉ thay đổi tên máy chủ *tính* của máy cục bộ thành `firewall`?

Với tùy chọn `--static`: `hostnamectl --static set-hostname fire`.

2. Những chi tiết nào ngoài tên máy chủ có thể được sửa đổi bằng lệnh `hostnamectl`?

`hostnamectl` cũng có thể đặt biểu tượng mặc định cho máy cục bộ, loại khung máy, vị trí và môi trường triển khai.

3. Mục nào trong `/etc/hosts` sẽ liên kết cả hai tên `firewall` và `router` với IP `10.8.0.1`?

Dòng `10.8.0.1 firewall router`.

4. Làm cách nào để có thể sửa đổi tệp `/etc/resolv.conf` để gửi tất cả các yêu cầu DNS đến `1.1.1.1`?

Sử dụng `nameserver 1.1.1.1` làm mục nhập máy chủ tên miền duy nhất.



109.2 Bài 2

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	109 Những yếu tố cơ bản về Mạng
Mục tiêu:	109.2 Cấu hình Mạng bền vững
Bài:	2 trên 2

Giới thiệu

Linux hỗ trợ hầu hết mọi công nghệ mạng được sử dụng để kết nối máy chủ, bộ chứa, máy ảo, máy tính và các thiết bị di động. Các kết nối giữa tất cả các nút mạng này có thể là các kết nối động và không đồng nhất, vì vậy mà chúng đòi hỏi một hệ thống quản lý thích hợp được cung cấp bởi hệ điều hành chạy bên trong.

Trước đây, các nhà phân phối đã phát triển các giải pháp tùy chỉnh của riêng họ để quản lý cơ sở hạ tầng mạng động. Ngày nay, các công cụ như *NetworkManager* và *systemd* đã cung cấp các tính năng tích hợp cao và toàn diện hơn để đáp ứng nhiều nhu cầu cụ thể.

NetworkManager

Hầu hết các bản phân phối Linux đều sử dụng trình nền dịch vụ *NetworkManager* để định cấu hình và kiểm soát các kết nối mạng của hệ thống. Mục đích của *NetworkManager* là làm cho cấu hình mạng trở nên đơn giản và tự động nhất có thể. Ví dụ: khi sử dụng DHCP, *NetworkManager* sẽ sắp xếp các thay đổi về tuyến, tìm nạp địa chỉ IP và cập nhật danh sách máy chủ DNS cục bộ nếu cần. Khi có cả kết nối có dây và không dây, *NetworkManager* sẽ ưu tiên kết nối có dây theo mặc định. *NetworkManager*, bất cứ khi nào có thể, sẽ cố gắng duy trì ít nhất một kết nối luôn hoạt

động.

NOTE

Một yêu cầu sử dụng DHCP (*Giao thức cấu hình máy chủ động*) thường được gửi qua bộ điều hợp mạng ngay khi liên kết tới mạng được thiết lập. Sau đó, máy chủ DHCP đang hoạt động trên mạng sẽ phản hồi bằng các cài đặt (địa chỉ IP, mặt nạ mạng, tuyến mặc định, v.v.) mà người yêu cầu phải sử dụng để liên lạc qua giao thức IP.

Theo mặc định, trình nền NetworkManager sẽ kiểm soát các giao diện mạng không được đề cập tới trong tệp `/etc/network/interfaces`. Nó làm như vậy để không can thiệp vào các phương pháp cấu hình khác có thể có, từ đó chỉ sửa đổi các giao diện không được giám sát.

Dịch vụ NetworkManager chạy ở chế độ nền với quyền gốc và sẽ kích hoạt các hành động cần thiết để duy trì hệ thống trực tuyến. Người dùng thông thường có thể tạo và sửa đổi các kết nối mạng với các ứng dụng khách; dù bản thân họ không có quyền gốc nhưng họ vẫn sẽ có khả năng giao tiếp với dịch vụ cơ bản để thực hiện các hành động được yêu cầu.

Các ứng dụng khách dành cho NetworkManager có sẵn cho cả dòng lệnh và môi trường đồ họa. Đối với môi trường đồ họa, ứng dụng khách sẽ xuất hiện dưới dạng một chương trình phụ trợ của môi trường máy tính (dưới các tên như *nm-tray*, *network-manager-gnome*, *nm-applet* hoặc *plasma-nm*) và nó thường có thể được truy cập thông qua một biểu tượng ở góc của thanh máy tính hoặc từ tiện ích cấu hình hệ thống.

Trong dòng lệnh, bản thân NetworkManager sẽ tự cung cấp hai chương trình máy khách là `nmcli` và `nmtui`. Cả hai chương trình này đều có các tính năng cơ bản giống nhau, nhưng `nmtui` có giao diện dựa trên thư viện lập trình curses, trong khi `nmcli` lại là một lệnh toàn diện hơn cũng có thể được sử dụng trong tệp lệnh. Lệnh `nmcli` sẽ chia tất cả các đặc tính liên quan đến mạng do NetworkManager kiểm soát thành các danh mục được gọi là *đối tượng* (objects):

general

Trạng thái và các hoạt động tổng quát của NetworkManager.

networking

Kiểm soát mạng tổng thể.

radio

Bộ chuyển mạch vô tuyến của NetworkManager.

connection

Các kết nối của NetworkManager.

device

Các thiết bị được quản lý bởi NetworkManager.

agent

Tác nhân bí mật hoặc tác nhân polkit của NetworkManager.

monitor

Giám sát các thay đổi của NetworkManager.

Tên đối tượng là đối số chính của lệnh `nmcli`. Ví dụ: để hiển thị trạng thái kết nối tổng thể của hệ thống, đối tượng `general` phải được đưa ra làm đối số:

```
$ nmcli general
STATE      CONNECTIVITY  WIFI-HW  WIFI      WWAN-HW  WWAN
connected  full          enabled  enabled   enabled  enabled
```





Cột STATE sẽ cho biết hệ thống có được kết nối với mạng hay không. Nếu kết nối bị hạn chế do cấu hình sai từ bên ngoài hoặc hạn chế truy cập thì cột CONNECTIVITY sẽ không báo cáo trạng thái kết nối là full. Nếu Portal xuất hiện trong cột CONNECTIVITY thì tức là cần phải có các bước xác thực bổ sung (thường thông qua trình duyệt web) để hoàn tất quá trình kết nối. Các cột còn lại sẽ báo cáo trạng thái của các kết nối không dây (nếu có), WIFI hoặc WWAN (Mạng diện rộng không dây - tức mạng di động). Hậu tố HW cho biết trạng thái tương ứng với thiết bị mạng chứ không phải kết nối mạng hệ thống - tức phần cứng đang được kích hoạt hay vô hiệu hoá để tiết kiệm điện.

Ngoài đối số đối tượng, `nmcli` cũng cần một đối số lệnh để thực thi. Lệnh `status` sẽ được sử dụng theo mặc định nếu không có đối số lệnh nào, do đó nên lệnh `nmcli General` thực sự nên được hiểu là `nmcli General status`.

Chúng ta hầu như không cần thực hiện bất kỳ một hành động nào khi bộ điều hợp mạng được kết nối trực tiếp với điểm truy cập thông qua cáp, nhưng mạng không dây sẽ yêu cầu các tương tác ở mức cao hơn để chấp nhận các thành viên mới. `nmcli` sẽ tạo điều kiện thuận lợi cho quá trình kết nối và sẽ lưu cài đặt để tự động kết nối trong tương lai, do đó nên nó rất hữu ích cho máy tính xách tay hoặc bất kỳ một thiết bị di động nào.

Trước khi kết nối với wi-fi, chúng ta trước tiên nên liệt kê các mạng khả dụng trong khu vực. Nếu hệ thống có một bộ điều hợp wi-fi đang hoạt động thì đối tượng `device` sẽ sử dụng nó để quét các mạng khả dụng bằng lệnh `nmcli device wifi list`:

```
$ nmcli device wifi list
IN-USE  BSSID          SSID      MODE  CHAN  RATE          SIGNAL  BARS  SECURITY
-----
        90:F6:52:C5:FA:12  Hypnotoad  Infra  11    130 Mbit/s    67      ████  WPA2
```

10:72:23:C7:27:AC	Jumbao	Infra	1	130 Mbit/s	55		WPA2
00:1F:33:33:E9:BE	NETGEAR	Infra	1	54 Mbit/s	35		WPA1 WPA2
A4:33:D7:85:6D:B0	AP53	Infra	11	130 Mbit/s	32		WPA1 WPA2
98:1E:19:1D:CC:3A	Bruma	Infra	1	195 Mbit/s	22		WPA1 WPA2

Hầu hết mọi người dùng đều sẽ sử dụng tên trong cột SSID để xác định mạng họ cần. Ví dụ: lệnh `nmcli` có thể kết nối với mạng có tên Hypnotoad bằng cách sử dụng lại đối tượng `device`:

```
$ nmcli device wifi connect Hypnotoad
```

Nếu lệnh được thực thi bên trong một trình mô phỏng cửa sổ dòng lệnh trong môi trường đồ họa thì một hộp thoại sẽ xuất hiện để yêu cầu cụm mật khẩu của mạng. Khi được thực thi trong bảng điều khiển thuần văn bản, mật khẩu có thể được cung cấp cùng với các đối số khác:

```
$ nmcli device wifi connect Hypnotoad password MyPassword
```

Nếu mạng wi-fi ẩn tên SSID của nó, `nmcli` vẫn có thể kết nối với mạng đó bằng các đối số `hidden yes` bổ sung:

```
$ nmcli device wifi connect Hypnotoad password MyPassword hidden yes
```

Nếu hệ thống có nhiều bộ điều hợp wi-fi, bộ điều hợp được sử dụng có thể được biểu thị bằng `ifname`. Ví dụ: để kết nối bằng bộ điều hợp có tên `wlo1`:

```
$ nmcli device wifi connect Hypnotoad password MyPassword ifname wlo1
```

Sau khi kết nối thành công, NetworkManager sẽ đặt tên nó theo SSID tương ứng (nếu đó là kết nối wi-fi) và sẽ giữ nó lại cho các kết nối trong tương lai. Tên kết nối và UUID của chúng có thể được liệt kê bằng lệnh `nmcli Connection show`:

```
$ nmcli connection show
NAME                UUID                                TYPE      DEVICE
Ethernet            53440255-567e-300d-9922-b28f0786f56e ethernet enp3s5
tun0                cae685e1-b0c4-405a-8ece-6d424e1fb5f8 tun       tun0
Hypnotoad           6fdec048-bcc5-490a-832b-da83d8cb7915 wifi      wlo1
4G                  a2cf4460-0cb7-42e3-8df3-ccb927f2fd88 gsm       --
```

Loại của mỗi kết nối cũng sẽ được hiển thị—có thể là `ethernet`, `wifi`, `tun`, `gsm`, `bridge`,

v.v. — cũng như thiết bị mà chúng được liên kết tới. Để thực hiện các hành động trên một kết nối cụ thể, ta phải cung cấp tên hoặc UUID của chúng. Ví dụ: để tắt kết nối Hypnotoad:

```
$ nmcli connection down Hypnotoad
Connection 'Hypnotoad' successfully deactivated
```

Tương tự như vậy, lệnh `nmcli connect up Hypnotoad` cũng có thể được sử dụng để khởi động kết nối vì giờ đây nó đã được NetworkManager lưu lại. Tên giao diện cũng có thể được sử dụng để kết nối lại nhưng trong trường hợp này ta nên sử dụng đối tượng `device`:

```
$ nmcli device disconnect wlo2
Device 'wlo1' successfully disconnected.
```

Tên giao diện cũng có thể được sử dụng để thiết lập lại kết nối:

```
$ nmcli device connect wlo2
Device 'wlo1' successfully activated with '833692de-377e-4f91-a3dc-d9a2b1fcf6cb'.
```

Hãy lưu ý rằng UUID kết nối sẽ thay đổi mỗi khi kết nối được kích hoạt nên chúng ta nên sử dụng tên của nó để đảm bảo được tính nhất quán.

Nếu có sẵn bộ điều hợp không dây đang không sử dụng tới thì chúng ta có thể tắt nó để tiết kiệm điện. Lần này, đối tượng `radio` phải được truyền tới `nmcli`:

```
$ nmcli radio wifi off
```

Tất nhiên, thiết bị không dây có thể được bật lại bằng lệnh `nmcli radio wifi on`.

Sau khi kết nối được thiết lập, chúng ta sẽ không cần phải tương tác thủ công trong tương lai nữa vì NetworkManager sẽ xác định các mạng đã biết có sẵn và tự động kết nối với chúng. Nếu cần, NetworkManager sẽ có các trình cắm có thể mở rộng chức năng của nó (chẳng hạn như trình cắm hỗ trợ kết nối VPN).

systemd-networkd

Các hệ thống chạy systemd có thể tùy ý sử dụng các trình nền tích hợp sẵn của nó để quản lý kết nối mạng: `systemd-networkd` để kiểm soát các giao diện mạng và `systemd-resolved` để quản lý việc phân giải tên cục bộ. Các dịch vụ này có thể tương thích ngược với các phương pháp cấu hình Linux cũ, nhưng riêng về cấu hình của giao diện mạng chúng cũng có các tính năng rất đáng để

biết đến.

Chúng ta có thể tìm thấy các tệp cấu hình được `systemd-networkd` sử dụng để thiết lập giao diện mạng trong bất kỳ một thư mục nào trong ba thư mục sau:

`/lib/systemd/network`

Thư mục mạng hệ thống.

`/run/systemd/network`

Thư mục mạng thời gian chạy khả biến.

`/etc/systemd/network`

Thư mục mạng quản trị cục bộ.

Các tệp sẽ được xử lý theo thứ tự từ điển nên tên của chúng nên được bắt đầu bằng các số để dễ đọc và đặt thứ tự hơn.

Các tệp trong `/etc` có mức độ ưu tiên cao nhất trong khi các tệp trong `/run` sẽ được ưu tiên hơn các tệp có cùng tên trong `/lib`. Điều này có nghĩa là nếu các tệp cấu hình trong các thư mục khác nhau có cùng tên thì `systemd-networkd` sẽ bỏ qua các tệp có mức độ ưu tiên thấp hơn. Việc tách các tệp như vậy là một cách để thay đổi cài đặt giao diện mà không cần phải sửa đổi các tệp gốc: các sửa đổi có thể được đặt trong `/etc/systemd/network` để ghi đè những cài đặt đó trong `/lib/systemd/network`.

Mục đích của mỗi tệp cấu hình sẽ phụ thuộc vào hậu tố của nó. Tên tệp kết thúc bằng `.netdev` sẽ được `systemd-networkd` sử dụng để tạo các thiết bị mạng ảo như thiết bị *bridge* hoặc *tun*. Các tệp kết thúc bằng `.link` sẽ đặt cấu hình cấp thấp cho giao diện mạng tương ứng. `systemd-networkd` sẽ tự động phát hiện và định cấu hình các thiết bị mạng khi chúng xuất hiện — cũng như bỏ qua các thiết bị đã được định cấu hình bằng các phương tiện khác — vì vậy nên trong hầu hết các trường hợp chúng ta sẽ không cần phải thêm các tệp này

Hậu tố quan trọng nhất là `.network`. Các tệp sử dụng hậu tố này có thể được sử dụng để thiết lập địa chỉ và tuyến mạng. Giống như các loại tệp cấu hình khác, tên của tệp sẽ xác định thứ tự tệp sẽ được xử lý. Giao diện mạng mà tệp cấu hình tham chiếu đến sẽ được xác định trong phần `[Match]` bên trong tệp.

Ví dụ: chúng ta có thể chọn giao diện mạng ethernet `enp3s5` trong tệp `/etc/systemd/network/30-lan.network` bằng cách sử dụng mục nhập `Name=enp3s5` trong phần `[Match]`:

```
[Match]
```

```
Name=enp3s5
```

Một danh sách các tên được phân tách bằng khoảng trắng cũng sẽ được chấp nhận để khớp với nhiều giao diện mạng với cùng một tệp này trong cùng một lúc. Tên có thể chứa các khối khớp kiểu vỏ như `en*`. Các mục khác sẽ cung cấp các quy tắc khớp khác nhau như chọn thiết bị mạng theo địa chỉ MAC của nó:

```
[Match]
MACAddress=00:16:3e:8d:2b:5b
```

Các cài đặt cho thiết bị sẽ nằm trong phần `[Network]` của tệp. Một cấu hình mạng tĩnh đơn giản sẽ chỉ yêu cầu các mục `Address` (Địa chỉ) và `Gateway` (Cổng):

```
[Match]
MACAddress=00:16:3e:8d:2b:5b

[Network]
Address=192.168.0.100/24
Gateway=192.168.0.1
```

Để sử dụng giao thức DHCP thay vì địa chỉ IP tĩnh, chúng ta nên sử dụng mục `DHCP`:

```
[Match]
MACAddress=00:16:3e:8d:2b:5b

[Network]
DHCP=yes
```

Dịch vụ `systemd-networkd` sẽ cố gắng tìm nạp cả địa chỉ IPv4 và IPv6 cho giao diện mạng. Để chỉ sử dụng IPv4, chúng ta nên sử dụng `DHCP=ipv4`. Tương tự, `DHCP=ipv6` sẽ bỏ qua cài đặt IPv4 và chỉ sử dụng địa chỉ IPv6 được cung cấp.

Các mạng không dây được bảo vệ bằng mật khẩu cũng có thể được cấu hình bởi `systemd-networkd` nhưng bộ điều hợp mạng phải được xác thực trong mạng trước khi `systemd-networkd` có thể định cấu hình nó. Việc xác thực được thực hiện bởi `WPA supplicant` - một chương trình dành riêng để định cấu hình bộ điều hợp mạng cho các mạng được bảo vệ bằng mật khẩu.

Bước đầu tiên là tạo tệp thông tin đăng nhập bằng lệnh `wpa_passphrase`:

```
# wpa_passphrase MyWifi > /etc/wpa_supplicant/wpa_supplicant-wlo1.conf
```

Lệnh này sẽ lấy cụm mật khẩu cho mạng không dây MyWifi từ đầu vào tiêu chuẩn và lưu trữ hàm băm của nó trong `/etc/wpa_supplicant/wpa_supplicant-wlo1.conf`. Hãy lưu ý rằng tên tệp phải chứa tên thích hợp của giao diện không dây (do đó mà tên tệp có chứa `wlo1`).

Trình quản lý `systemd` sẽ đọc các tệp cụm mật khẩu WPA trong `/etc/wpa_supplicant/` và tạo dịch vụ tương ứng để chạy WPA supplicant và đưa giao diện lên. Khi đó, tệp cụm mật khẩu được tạo trong ví dụ sẽ có đơn vị dịch vụ tương ứng được gọi là `wpa_supplicant@wlo1.service`. Lệnh `systemctl start wpa_supplicant@wlo1.service` sẽ liên kết bộ điều hợp không dây với điểm truy cập từ xa. Lệnh `systemctl Enable wpa_supplicant@wlo1.service` sẽ làm cho việc liên kết trở thành tự động trong thời gian khởi động.

Cuối cùng, tệp `.network` khớp với giao diện `wlo1` phải hiện hữu trong `/etc/systemd/network/` vì `systemd-networkd` sẽ sử dụng nó để định cấu hình giao diện ngay khi WPA supplicant kết thúc liên kết với điểm truy cập.

Bài tập Hướng dẫn

1. Ý nghĩa của từ `Portal` trong cột `CONNECTIVITY` trong đầu ra của lệnh `nmcli General status` là gì?

2. Trong một cửa sổ dòng lệnh bảng điều khiển, làm cách nào để một người dùng thông thường có thể sử dụng lệnh `nmcli` để kết nối với mạng không dây `MyWifi` được bảo vệ bằng mật khẩu `MyPassword`?

3. Lệnh nào có thể bật bộ điều hợp không dây nếu trước đó nó đã bị hệ điều hành tắt đi?

4. Các tệp cấu hình tùy chỉnh nên được đặt trong thư mục nào khi `systemd-networkd` đang quản lý giao diện mạng?

Bài tập Mở rộng

1. Làm cách nào để một người dùng có thể chạy lệnh `nmcli` để xóa một kết nối không sử dụng tới có tên `Hotel Internet`?

2. `NetworkManager` sẽ quét mạng wi-fi định kỳ và lệnh `nmcli device wifi list` sẽ chỉ liệt kê các điểm truy cập được tìm thấy trong lần quét cuối cùng. Lệnh `nmcli` nên được sử dụng như thế nào để yêu cầu `NetworkManager` quét lại ngay lập tức tất cả các điểm truy cập có sẵn?

3. Nên sử dụng mục `name` nào trong phần `[Match]` của tệp cấu hình `systemd-networkd` để khớp với tất cả các giao diện ethernet?

4. Lệnh `wpa_passphrase` nên được thực thi như thế nào để sử dụng cụm mật khẩu làm đối số mặc định chứ không phải từ đầu vào tiêu chuẩn?

Tóm tắt

Bài học này đã đề cập đến các công cụ phổ biến được sử dụng trong Linux để quản lý các kết nối mạng động và không đồng nhất. Mặc dù hầu hết các phương pháp cấu hình không yêu cầu sự can thiệp của người dùng nhưng đôi khi điều đó lại là cần thiết và các công cụ như *NetworkManager* và *systemd-networkd* có thể giảm thiểu các rắc rối đến mức tối thiểu. Bài học đã đi qua các chủ đề sau:

- Cách NetworkManager và systemd-networkd tích hợp với hệ thống.
- Cách người dùng có thể tương tác với NetworkManager và systemd-networkd.
- Cấu hình giao diện cơ bản với cả NetworkManager và systemd-networkd.

Các khái niệm, lệnh và quy trình đã được đề cập tới là:

- Các lệnh máy khách của NetworkManager: `nmtui` và `nmcli`.
- Quét và kết nối với mạng không dây bằng các lệnh `nmcli` thích hợp.
- Kết nối mạng wi-fi bền vững bằng cách sử dụng `systemd-networkd`.

Đáp án Bài tập Hướng dẫn

1. Ý nghĩa của từ `Portal` trong cột `CONNECTIVITY` trong đầu ra của lệnh `nmcli General status` là gì?

Nó có nghĩa là chúng ta cần phải có các bước xác thực bổ sung (thường thông qua trình duyệt web) để hoàn tất quá trình kết nối.

2. Trong một cửa sổ dòng lệnh bảng điều khiển, làm cách nào để một người dùng thông thường có thể sử dụng lệnh `nmcli` để kết nối với mạng không dây `MyWifi` được bảo vệ bằng mật khẩu `MyPassword`?

Trong một cửa sổ dòng lệnh thuần văn bản, lệnh sẽ là:

```
$ nmcli device wifi connect MyWifi password MyPassword
```

3. Lệnh nào có thể bật bộ điều hợp không dây nếu trước đó nó đã bị hệ điều hành tắt đi?

```
$ nmcli radio wifi on
```

4. Các tệp cấu hình tùy chỉnh nên được đặt trong thư mục nào khi `systemd-networkd` đang quản lý giao diện mạng?

Trong thư mục mạng quản trị cục bộ: `/etc/systemd/network`.

Đáp án Bài tập Mở rộng

1. Làm cách nào để một người dùng có thể chạy lệnh `nmcli` để xóa một kết nối không sử dụng tới có tên `Hotel Internet`?

```
$ nmcli connection delete "Hotel Internet"
```

2. NetworkManager sẽ quét mạng wi-fi định kỳ và lệnh `nmcli device wifi list` sẽ chỉ liệt kê các điểm truy cập được tìm thấy trong lần quét cuối cùng. Lệnh `nmcli` nên được sử dụng như thế nào để yêu cầu NetworkManager quét lại ngay lập tức tất cả các điểm truy cập có sẵn?

Siêu người dùng có thể chạy `nmcli device wifi rescan` để yêu cầu NetworkManager quét lại các điểm truy cập khả dụng.

3. Nên sử dụng mục `name` nào trong phần `[Match]` của tệp cấu hình `systemd-networkd` để khớp với tất cả các giao diện ethernet?

Mục nhập `name=en*` vì `en` là tiền tố cho giao diện ethernet trong Linux và `systemd-networkd` sẽ chấp nhận các khối khớp mẫu giống vỏ.

4. Lệnh `wpa_passphrase` nên được thực thi như thế nào để sử dụng cụm mật khẩu làm đối số mặc định chứ không phải từ đầu vào tiêu chuẩn?

Mật khẩu phải được đặt ngay sau SSID như trong `wpa_passphrase MyWifi MyPassword`.



109.3 Xử lý sự cố Mạng cơ bản

Tham khảo các mục tiêu LPI

LPIC-1 version 5.0, Exam 102, Objective 109.3

Khối lượng

4

Các lĩnh vực kiến thức chính

- Cấu hình thủ công các giao diện mạng (bao gồm việc xem và thay đổi cấu hình các giao diện mạng bằng iproute2).
- Định cấu hình định tuyến theo cách thủ công (bao gồm việc xem và thay đổi bảng định tuyến cũng như đặt tuyến mặc định bằng iproute2).
- Gỡ lỗi các vấn đề liên quan đến cấu hình mạng.
- Kiến thức về các lệnh công cụ mạng kế thừa.

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- ip
- hostname
- ss
- ping
- ping6
- traceroute
- traceroute6
- tracepath
- tracepath6

- netcat
- ifconfig
- netstat
- route



109.3 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	109 Những yếu tố cơ bản về Mạng
Mục tiêu:	109.3 Khắc phục sự cố Mạng cơ bản
Bài:	1 trên 2

Giới thiệu

Linux có khả năng hoạt động mạng rất linh hoạt và mạnh mẽ. Trên thực tế, hệ điều hành dựa trên Linux thường được sử dụng trên các thiết bị mạng phổ thông bao gồm cả thiết bị thương mại có chi phí cao. Riêng bản thân việc vận hành mạng của Linux cũng là một chứng chỉ. Từ cốt lõi như vậy, bài học này sẽ chỉ đề cập đến một số công cụ xử lý sự cố và cấu hình cơ bản.

Hãy xem lại các bài học về giao thức internet và cấu hình mạng bền vững trước bài học này. Trong bài học này, chúng ta sẽ tìm hiểu về các công cụ để định cấu hình và khắc phục sự cố mạng IPv4 và IPv6.

Mặc dù không phải là mục tiêu chính thức, các *trình thám thính gói* như `tcpdump` là những công cụ khắc phục sự cố rất hữu ích. Trình thám thính gói cho phép người dùng xem và ghi lại các gói đi vào hoặc ra khỏi giao diện mạng. Các công cụ như *trình xem thập phân* (hex viewer) và *trình phân tích giao thức* (protocol analyzer) có thể được sử dụng để xem các gói này một cách chi tiết hơn so với mức độ mà trình thám thính gói thường cho phép. Người dùng được khuyến khích tìm hiểu và biết rõ về các chương trình này.

Giới thiệu về lệnh ip

Lệnh `ip` là một tiện ích khá mới được sử dụng để xem và định cấu hình mọi thứ liên quan đến cấu hình mạng. Bài học này sẽ đề cập đến một số lệnh con được sử dụng nhiều nhất; tuy nhiên, chúng cũng vẫn chỉ là bề nổi của `ip`. Việc học cách đọc tài liệu sẽ giúp chúng ta sử dụng nó hiệu quả hơn rất nhiều.

Mỗi lệnh con của `ip` đều có trang hướng dẫn riêng của nó. Phần XEM CSONG của trang hướng dẫn của `ip` có một danh sách về các lệnh con này:

```
$ man ip
...
SEE ALSO
    ip-address(8), ip-adrlabel(8), ip-l2tp(8), ip-link(8), ip-maddress(8),
    ip-monitor(8), ip-mroute(8), ip-neighbour(8), ip-netns(8), ip-
    ntable(8), ip-route(8), ip-rule(8), ip-tcp_metrics(8), ip-token(8), ip-
    tunnel(8), ip-xfrm(8)
    IP Command reference ip-cref.ps
...
```

Thay vì phải xem toàn bộ phần này khi cần tới trang hướng dẫn, chúng ta chỉ cần thêm `-` và tên của lệnh con vào `ip` (ví dụ như `man ip-route`).

Một nguồn thông tin khác là chức năng trợ giúp. Để xem phần trợ giúp được tích hợp sẵn, hãy thêm `help` sau lệnh con:

```
$ ip address help
Usage: ip address {add|change|replace} IFADDR dev IFNAME [ LIFETIME ]
                                     [ CONFFLAG-LIST ]

    ip address del IFADDR dev IFNAME [mngtmpaddr]
    ip address {save|flush} [ dev IFNAME ] [ scope SCOPE-ID ]
                                     [ to PREFIX ] [ FLAG-LIST ] [ label LABEL ] [up]
    ip address [ show [ dev IFNAME ] [ scope SCOPE-ID ] [ master DEVICE ]
                                     [ type TYPE ] [ to PREFIX ] [ FLAG-LIST ]
                                     [ label LABEL ] [up] [ vrf NAME ] ]
    ip address {showdump|restore}
IFADDR := PREFIX | ADDR peer PREFIX
...
```

Nhắc lại về Mặt nạ Mạng và Định Tuyến

IPv4 và IPv6 là những giao thức được định tuyến hoặc có thể định tuyến. Điều này có nghĩa là chúng được thiết kế để giúp các nhà thiết kế mạng có thể kiểm soát luồng lưu lượng. Ethernet không phải là một giao thức có thể định tuyến - tức là nếu kết nối nhiều thiết bị với nhau mà không sử dụng gì ngoài Ethernet, người dùng sẽ khó có thể kiểm soát luồng lưu lượng mạng. Bất kỳ một biện pháp kiểm soát lưu lượng truy cập nào cũng sẽ kết thúc tương tự như các giao thức định tuyến và có thể định tuyến hiện tại.

Các giao thức có thể định tuyến cho phép nhà thiết kế mạng phân đoạn các mạng để giảm yêu cầu xử lý của các thiết bị kết nối cũng như cung cấp các tính năng dự phòng và quản lý lưu lượng.

Địa chỉ IPv4 và IPv6 có hai phần. Bộ bit đầu tiên sẽ tạo nên phần mạng, trong khi bộ thứ hai sẽ tạo nên phần máy chủ. Số bit tạo nên phần mạng được xác định bởi *mặt nạ mạng* (còn được gọi là *mặt nạ mạng con*). Đôi khi nó còn được gọi là *độ dài tiền tố*. Dù được gọi là gì, nó vẫn chính là số bit mà máy coi là phần mạng của địa chỉ. Với IPv4, đôi khi điều này sẽ được chỉ định bằng ký hiệu thập phân có dấu chấm.

Dưới đây là một ví dụ sử dụng IPv4. Hãy lưu ý cách các chữ số nhị phân duy trì giá trị vị trí của chúng trong các 8 bit ngay cả khi mặt nạ mạng chia nó ra.

```
192.168.130.5/20
```

```
      192      168      130      5
      11000000 10101000 10000010 00000101
```

```
20 bits = 11111111 11111111 11110000 00000000
```

```
Network = 192.168.128.0
```

```
Host    = 2.5
```

Phần mạng của địa chỉ được máy IPv4 hoặc IPv6 sử dụng để tra cứu giao diện nào mà gói sẽ được gửi đi trong bảng định tuyến của nó. Khi máy chủ IPv4 hoặc IPv6 có kích hoạt định tuyến nhận được gói không dành cho chính máy chủ đó, nó sẽ cố gắng khớp phần mạng của điểm đích với mạng trong bảng định tuyến. Nếu tìm thấy một mục nhập khớp, nó sẽ gửi gói đến điểm đích được chỉ định trong bảng định tuyến. Nếu không tìm thấy mục nào và tuyến mặc định được định cấu hình, nó sẽ được gửi đến tuyến mặc định. Nếu không tìm thấy mục nào và không có tuyến mặc định nào được cấu hình thì gói sẽ bị loại bỏ.

Cấu hình Giao diện

Có hai công cụ chúng ta sẽ cùng tìm hiểu và sử dụng để định cấu hình giao diện mạng: `ifconfig` và `ip`. Mặc dù vẫn được sử dụng rộng rãi, chương trình `ifconfig` lại được coi là một công cụ cũ và có thể sẽ không có sẵn trên các hệ thống hiện đại.

TIP Trên các bản phân phối Linux mới hơn, việc cài đặt gói `net-tools` sẽ cung cấp cho người dùng các lệnh kết nối mạng cũ.

Trước khi định cấu hình một giao diện, chúng ta phải biết những giao diện nào đang có sẵn. Có một số cách để thực hiện điều này. Một trong số đó là sử dụng tùy chọn `-a` của `ifconfig`:

```
$ ifconfig -a
```

Một cách khác là với `ip`. Đôi khi chúng ta sẽ thấy các ví dụ có `ip addr`, `ip a` và `ip address`. Chúng đều tương đương với nhau nhưng lệnh con chính thức là `ip address`. Điều này có nghĩa là nếu muốn xem trang hướng dẫn, chúng ta phải sử dụng `man ip-address` chứ không phải `man ip-addr`.

Lệnh con `link` cho `ip` sẽ liệt kê các liên kết giao diện có sẵn để cấu hình:

```
$ ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
    link/ether 08:00:27:54:18:57 brd ff:ff:ff:ff:ff:ff
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
    link/ether 08:00:27:ab:11:3e brd ff:ff:ff:ff:ff:ff
```

Giả sử hệ thống tệp `sys` được gắn kết, chúng ta cũng có thể liệt kê nội dung của `/sys/class/net`:

```
$ ls /sys/class/net
enp0s3 enp0s8 lo
```

Để định cấu hình giao diện với `ifconfig`, chúng ta phải đăng nhập dưới tên siêu người dùng hoặc sử dụng tiện ích `sudo` để chạy lệnh với đặc quyền gốc. Hãy thực hiện theo ví dụ dưới đây:

```
# ifconfig enp1s0 192.168.50.50/24
```

Phiên bản Linux của `ifconfig` rất linh hoạt trong việc chỉ định mặt nạ mạng con:

```
# ifconfig eth2 192.168.50.50 netmask 255.255.255.0
# ifconfig eth2 192.168.50.50 netmask 0xfffff00
# ifconfig enp0s8 add 2001:db8::10/64
```

Hãy lưu ý cách sử dụng từ khóa `add` với IPv6. Nếu không thêm `add` vào trước địa chỉ IPv6, chúng ta sẽ nhận được thông báo lỗi.

Lệnh sau sẽ định cấu hình giao diện với `ip`:

```
# ip addr add 192.168.5.5/24 dev enp0s8
# ip addr add 2001:db8::10/64 dev enp0s8
```

Với `ip`, cả IPv4 và IPv6 đều sử dụng một lệnh giống nhau.

Định cấu hình Tùy chọn cấp thấp

Lệnh `ip link` được sử dụng để định cấu hình cài đặt giao thức hoặc giao diện cấp thấp như Vlan, ARP hoặc MTU hoặc để vô hiệu hóa một giao diện.

Một tác vụ rất phổ biến của `ip link` là tắt hoặc bật một giao diện. Điều này cũng có thể được thực hiện với `ifconfig`:

```
# ip link set dev enp0s8 down
# ip link show dev enp0s8
3: enp0s8: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast state DOWN mode DEFAULT group
default qlen 1000
    link/ether 08:00:27:ab:11:3e brd ff:ff:ff:ff:ff:ff
# ifconfig enp0s8 up
# ip link show dev enp0s8
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
    link/ether 08:00:27:ab:11:3e brd ff:ff:ff:ff:ff:ff
```

Đôi khi chúng ta có thể sẽ cần phải điều chỉnh MTU của giao diện. Giống như việc bật/tắt giao diện, điều này cũng có thể được thực hiện bằng `ifconfig` của `ip link`:


```
# ip link set enp0s8 mtu 2000
# ip link show dev enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 2000 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
    link/ether 08:00:27:54:53:59 brd ff:ff:ff:ff:ff:ff
# ifconfig enp0s3 mtu 1500
# ip link show dev enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
    link/ether 08:00:27:54:53:59 brd ff:ff:ff:ff:ff:ff
```

Bảng định tuyến

Các lệnh `route`, `netstat -r` và `ip Route` đều có thể được sử dụng để xem bảng định tuyến. Nếu muốn sửa đổi tuyến của mình, chúng ta cần sử dụng `route` hoặc `ip Route`. Dưới đây là các ví dụ về xem một bảng định tuyến:

```
$ netstat -r
Kernel IP routing table
Destination      Gateway          Genmask         Flags   MSS Window  irtt Iface
default          10.0.2.2        0.0.0.0         UG      0 0        0 enp0s3
10.0.2.0         0.0.0.0         255.255.255.0   U        0 0        0 enp0s3
192.168.150.0    0.0.0.0         255.255.255.0   U        0 0        0 enp0s8

$ ip route
default via 10.0.2.2 dev enp0s3 proto dhcp metric 100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.15 metric 100
192.168.150.0/24 dev enp0s8 proto kernel scope link src 192.168.150.200

$ route
Kernel IP routing table
Destination      Gateway          Genmask         Flags Metric Ref    Use Iface
default          10.0.2.2        0.0.0.0         UG    100    0      0 enp0s3
10.0.2.0         0.0.0.0         255.255.255.0   U     100    0      0 enp0s3
192.168.150.0    0.0.0.0         255.255.255.0   U      0      0      0 enp0s8
```

Hãy lưu ý việc không có đầu ra nào liên quan đến IPv6. Nếu muốn xem bảng định tuyến cho IPv6, chúng ta phải sử dụng `route -6`, `netstat -6r` và `ip -6 route`.

```
$ route -6
Kernel IPv6 routing table
Destination      Next Hop          Flag Met Ref Use If
2001:db8::/64    [::]             U   256 0   0 enp0s8
```

fe80::/64	:::	U	100	0	0	enp0s3
2002:a00::/24	:::	!n	1024	0	0	lo
:::]/0	2001:db8::1	UG	1	0	0	enp0s8
localhost/128	:::	Un	0	2	84	lo
2001:db8::10/128	:::	Un	0	1	0	lo
fe80::a00:27ff:fe54:5359/128	:::	Un	0	1	0	lo
ff00::/8	:::	U	256	1	3	enp0s3
ff00::/8	:::	U	256	1	6	enp0s8

Một ví dụ về `netstat -r6` đã bị bỏ qua vì đầu ra của nó giống hệt với `route -6`. Một số kết quả đầu ra của lệnh `route` ở trên đã tự giải thích ý nghĩa của chúng. Cột `Flag` đã cung cấp một số thông tin về tuyến. Cờ `U` biểu thị rằng có một tuyến đang hoạt động. `!` có nghĩa là từ chối tuyến - tức là tuyến có ký hiệu `!` sẽ không được sử dụng. Cờ `n` có nghĩa là tuyến chưa được lưu vào bộ nhớ đệm. Hạt nhân duy trì bộ nhớ đệm của các tuyến để tra cứu nhanh hơn và tách biệt với tất cả các tuyến đã biết. Cờ `G` biểu thị một cổng. Cột `Metric` hoặc `Met` không được hạt nhân sử dụng. Cột này cho biết khoảng cách quản trị đến mục tiêu. Khoảng cách quản trị này được các giao thức định tuyến sử dụng để xác định các tuyến động. Cột `Ref` là số lượng tham chiếu hoặc số lần sử dụng của một tuyến. Giống như `Metric`, nó không được nhân Linux sử dụng. Cột `Use` hiển thị số lần tra cứu cho một tuyến.

Trong đầu ra của `netstat -r`, `MSS` cho biết kích thước phân đoạn tối đa cho các kết nối TCP qua tuyến đó. Cột `Window` hiển thị kích thước cửa sổ TCP mặc định. `irtt` hiển thị thời gian trễ tron vòng cho các gói trên tuyến này.

Đầu ra của `ip Route` và `ip -6 Route` có thể đọc như sau:

1. Đích đến.
2. Địa chỉ tùy chọn theo sau là giao diện.
3. Giao thức định tuyến được sử dụng để thêm tuyến.
4. Phạm vi của tuyến. Nếu điều này bị bỏ qua thì đó là phạm vi toàn cục hoặc một cổng.
5. Số liệu của tuyến. Điều này được các giao thức định tuyến động sử dụng để xác định chi phí (cost) của tuyến nhưng nó lại không được sử dụng bởi hầu hết các hệ thống.
6. Nếu đó là một tuyến IPv6, tuyến RFC4191 sẽ được ưu tiên.

Hãy cùng làm việc với một vài ví dụ để làm rõ điều này:

Ví dụ về IPv4

```
default via 10.0.2.2 dev enp0s3 proto dhcp metric 100
```

1. Đích đến là tuyến mặc định.
2. Địa chỉ cổng là `10.0.2.2` có thể được truy cập thông qua giao diện `enp0s3`.
3. Nó đã được DHCP thêm vào bảng định tuyến.
4. Phạm vi đã bị bỏ qua, vì vậy nó mang phạm vi toàn cục.
5. Tuyến có giá trị chi phí là `100`.
6. Không có tùy chọn tuyến IPv6.

Ví dụ về IPv6

```
fc0::/64 dev enp0s8 proto kernel metric 256 pref medium
```

1. Đích đến là `fc0::/64`.
2. Nó có thể được truy cập thông qua giao diện `enp0s8`.
3. Nó được hạt nhân tự động thêm vào.
4. Phạm vi đã bị bỏ qua, vì vậy nó mang phạm vi toàn cục.
5. Tuyến có giá trị chi phí là `256`.
6. Nó có tùy chọn IPv6 là `medium` (trung bình).

Quản lý Tuyến

Các tuyến có thể được quản lý bằng cách sử dụng `route` hoặc `ip Route`. Dưới đây là một ví dụ về cách thêm và xóa tuyến bằng lệnh `route`. Với `route`, chúng ta sẽ phải sử dụng tùy chọn `-6` cho IPv6:

```
# ping6 -c 2 2001:db8:1::20
connect: Network is unreachable
# route -6 add 2001:db8:1::/64 gw 2001:db8::3
# ping6 -c 2 2001:db8:1::20
PING 2001:db8:1::20(2001:db8:1::20) 56 data bytes
64 bytes from 2001:db8:1::20: icmp_seq=1 ttl=64 time=0.451 ms
64 bytes from 2001:db8:1::20: icmp_seq=2 ttl=64 time=0.438 ms
# route -6 del 2001:db8:1::/64 gw 2001:db8::3
# ping6 -c 2 2001:db8:1::20
connect: Network is unreachable
```

Dưới đây là một ví dụ tương tự khi sử dụng lệnh `ip Route`:

```
# ping6 -c 2 2001:db8:1:20
connect: Network is unreachable
# ip route add 2001:db8:1::/64 via 2001:db8::3
# ping6 -c 2 2001:db8:1:20
PING 2001:db8:1::20(2001:db8:1::20) 56 data bytes
64 bytes from 2001:db8:1::20: icmp_seq=2 ttl=64 time=0.529 ms
64 bytes from 2001:db8:1::20: icmp_seq=2 ttl=64 time=0.438 ms
# ip route del 2001:db8:1::/64 via 2001:db8::3
# ping6 -c 2 2001:db8:1:20
connect: Network is unreachable
```

Bài tập Hướng dẫn

1. Những lệnh nào có thể được sử dụng để liệt kê các giao diện mạng?

2. Bạn có thể tạm thời vô hiệu hóa một giao diện bằng cách nào? Bạn sẽ kích hoạt lại nó bằng cách nào?

3. Lựa chọn nào sau đây là mặt nạ mạng con hợp lý cho IPv4?

0.0.0.255	
255.0.255.0	
255.252.0.0	
/24	

4. Bạn có thể sử dụng lệnh nào để xác minh tuyến mặc định của mình?

5. Làm cách nào để có thể thêm một địa chỉ IP thứ hai vào giao diện?

"

Bài tập Mở rộng

1. Lệnh con nào của `ip` có thể được sử dụng để định cấu hình gắn thẻ vlan?

2. Bạn sẽ định cấu hình một tuyến mặc định như thế nào?

3. Làm cách nào để có thể có được thông tin chi tiết về lệnh `ip neighbor`? Điều gì sẽ xảy ra nếu bạn chạy riêng lẻ lệnh này?

4. Bạn sẽ sao lưu bảng định tuyến của mình bằng cách nào? Làm thế nào để có thể sẽ khôi phục từ đó?

5. Lệnh con `ip` nào có thể được dùng để định cấu hình các tùy chọn chặn vòng lặp?

Tóm tắt

Kết nối mạng thường được cấu hình bởi các tệp lệnh khởi động của hệ thống hoặc một trình trợ giúp như NetworkManager. Hầu hết các bản phân phối đều có các công cụ giúp người dùng chỉnh sửa các tệp cấu hình tệp lệnh khởi động. Hãy tham khảo tài liệu của bản phân phối để biết thêm chi tiết.

Việc cấu hình mạng theo cách thủ công cho phép chúng ta khắc phục sự cố hiệu quả hơn. Nó rất hữu ích trong các môi trường tối thiểu được sử dụng cho những việc như khôi phục từ bản sao lưu hoặc di chuyển sang phần cứng mới.

Các tiện ích được nhắc tới trong bài học này có nhiều chức năng hơn so với những gì đã được đề cập. Việc xem trang hướng dẫn của từng chức năng để làm quen với các tùy chọn có sẵn sẽ rất hữu ích đối với người dùng. Các lệnh `ss` và `ip` là cách thực hiện hiện đại, trong khi những lệnh còn lại đã được đề cập đến được coi là các công cụ cũ (tuy vẫn được sử dụng khá phổ biến).

Cách tốt nhất để làm quen với các công cụ được đề cập tới là thực hành chúng. Chỉ với một máy tính có dung lượng RAM khiêm tốn, chúng ta đã có thể thiết lập một phòng thí nghiệm mạng ảo sử dụng các máy ảo để thực hành. Chỉ cần tới ba máy ảo là đã đủ để chúng ta làm quen với các công cụ này.

Các lệnh được sử dụng trong bài học này bao gồm:

ifconfig

Tiện ích kế thừa được sử dụng để định cấu hình giao diện mạng và xem lại trạng thái của chúng.

ip

Tiện ích hiện đại và linh hoạt được sử dụng để định cấu hình giao diện mạng và xem trạng thái của chúng.

netstat

Lệnh kế thừa được sử dụng để xem các kết nối mạng hiện tại và thông tin tuyến.

route

Lệnh kế thừa được sử dụng để xem hoặc sửa đổi bảng định tuyến của hệ thống.

Đáp án Bài tập Hướng dẫn

1. Những lệnh nào có thể được sử dụng để liệt kê các giao diện mạng?

Bất kỳ lệnh nào dưới đây:

```
ip link, ifconfig -a hoặc ls /sys/class/net
```

2. Bạn có thể tạm thời vô hiệu hóa một giao diện bằng cách nào? Bạn sẽ kích hoạt lại nó bằng cách nào?

Bạn có thể sử dụng `ifconfig` hoặc `ip link`:

Sử dụng `ifconfig`:

```
$ ifconfig wlan1 down
$ ifconfig wlan1 up
```

Sử dụng `ip link`:

```
$ ip link set wlan1 down
$ ip link set wlan1 up
```

3. Lựa chọn nào sau đây là mặt nạ mạng con hợp lý cho IPv4?

- 255.252.0.0
- /24

Các mặt nạ còn lại không hợp lệ vì chúng không tách địa chỉ thành hai phần một cách rõ ràng: phần đầu tiên xác định mạng và phần thứ hai là máy chủ. Hầu hết các bit bên trái của mặt nạ sẽ luôn là 1 và các bit bên phải sẽ luôn là 0.

4. Bạn có thể sử dụng lệnh nào để xác minh tuyến mặc định của mình?

Bạn có thể sử dụng `route`, `netstat -r` hoặc `ip route`:

```
$ route
Kernel IP routing table
Destination      Gateway          Genmask         Flags Metric Ref    Use Iface
default          server          0.0.0.0         UG    600    0      0 wlan1
192.168.1.0     0.0.0.0        255.255.255.0  U    600    0      0 wlan1
```



```

$ netstat -r
Kernel IP routing table
Destination      Gateway          Genmask         Flags   MSS Window  irtt Iface
default          server           0.0.0.0         UG      0 0        0 wlan1
192.168.1.0      0.0.0.0         255.255.255.0  U      0 0        0 wlan1
$ ip route
default via 192.168.1.20 dev wlan1 proto static metric 600
192.168.1.0/24 dev wlan1 proto kernel scope link src 192.168.1.24 metric 600

```

5. Làm cách nào để có thể thêm một địa chỉ IP thứ hai vào giao diện?

Bạn có thể sử dụng `ip address` hoặc `ifconfig`. Hãy nhớ rằng `ifconfig` là một công cụ cũ:

```
$ ip addr add 172.16.15.16/16 dev enp0s9 label enp0s9:sub1
```

Lệnh `label enp0s9:sub1` đã thêm một bí danh vào `enp0s9`. Nếu không sử dụng `ifconfig` kế thừa, bạn có thể bỏ qua phần này. Nếu có sử dụng, lệnh vẫn sẽ hoạt động, nhưng địa chỉ bạn vừa thêm sẽ không hiển thị trong đầu ra của `ifconfig`.

Bạn cũng có thể sử dụng `ifconfig`:

```
$ ifconfig enp0s9:sub1 172.16.15.16/16
```

Đáp án Bài tập Mở rộng

1. Lệnh con nào của `ip` có thể được sử dụng để định cấu hình gắn thẻ vlan?

`ip link` có tùy chọn `vlan` có thể được sử dụng. Dưới đây là một ví dụ về gắn thẻ giao diện phụ với vlan 20.

```
# ip link add link enp0s9 name enp0s9.20 type vlan id 20
```

2. Bạn sẽ định cấu hình một tuyến mặc định như thế nào?

Sử dụng `route` hoặc `ip route`:

```
# route add default gw 192.168.1.1  
# ip route add default via 192.168.1.1
```

3. Làm cách nào để có thể có được thông tin chi tiết về lệnh `ip neighbour`? Điều gì sẽ xảy ra nếu bạn chạy riêng lẻ lệnh này?

Bằng cách đọc trang hướng dẫn:

```
$ man ip-neighbour
```

Nó sẽ hiển thị bộ đệm ARP của bạn:

```
$ ip neighbour  
10.0.2.2 dev enp0s3 lladdr 52:54:00:12:35:02 REACHABLE
```

4. Bạn sẽ sao lưu bảng định tuyến của mình bằng cách nào? Làm thế nào để có thể sẽ khôi phục từ đó?

Ví dụ dưới đây một ví dụ minh họa về việc sao lưu và khôi phục một bảng định tuyến:

```
# ip route save > /root/routes/route_backup  
# ip route restore < /root/routes/route_backup
```

5. Lệnh con `ip` nào có thể được dùng để định cấu hình các tùy chọn chặn vòng lặp?

Tương tự như quản lý cài đặt vlan, `ip link` có thể định cấu hình cây bao trùm bằng cách sử

dụng loại `bridge`. Ví dụ sau sẽ cho thấy việc thêm giao diện ảo có mức ưu tiên STP là 50:

```
# ip link add link enp0s9 name enp0s9.50 type bridge priority 50
```



109.3 Bài 2

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	109 Những yếu tố cơ bản về Mạng
Mục tiêu:	109.3 Khắc phục sự cố Mạng cơ bản
Bài:	2 trên 2

Giới thiệu

Hệ điều hành dựa trên Linux có nhiều công cụ để khắc phục sự cố mạng. Bài học này sẽ đề cập đến một số công cụ phổ biến hơn. Tại thời điểm này, chúng ta đã phải nắm được cơ bản về OSI hoặc các mô hình mạng phân lớp khác, địa chỉ IPv4 hoặc IPv6 cũng như những kiến thức cơ bản về định tuyến và chuyển mạch.

Cách tốt nhất để kiểm tra kết nối mạng là thử sử dụng ứng dụng. Khi chúng không hoạt động, có rất nhiều công cụ có sẵn để giúp chúng ta chẩn đoán sự cố.

Kiểm tra kết nối bằng ping

Các lệnh `ping` và `ping6` có thể được sử dụng để gửi yêu cầu phản hồi (echo request) ICMP tương ứng đến địa chỉ IPv4 hoặc IPv6. Yêu cầu phản hồi ICMP sẽ gửi một lượng nhỏ dữ liệu đến địa chỉ đích. Nếu có thể truy cập địa chỉ đích, nó sẽ gửi tin nhắn trả lời phản hồi (echo reply) ICMP lại cho người gửi với cùng dữ liệu đã được gửi tới nó:

```
$ ping -c 3 192.168.50.2
PING 192.168.50.2 (192.168.50.2) 56(84) bytes of data.
```

```
64 bytes from 192.168.50.2: icmp_seq=1 ttl=64 time=0.525 ms
64 bytes from 192.168.50.2: icmp_seq=2 ttl=64 time=0.419 ms
64 bytes from 192.168.50.2: icmp_seq=3 ttl=64 time=0.449 ms

--- 192.168.50.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2006ms
rtt min/avg/max/mdev = 0.419/0.464/0.525/0.047 ms
```

```
$ ping6 -c 3 2001:db8::10
PING 2001:db8::10(2001:db8::10) 56 data bytes
64 bytes from 2001:db8::10: icmp_seq=1 ttl=64 time=0.425 ms
64 bytes from 2001:db8::10: icmp_seq=2 ttl=64 time=0.480 ms
64 bytes from 2001:db8::10: icmp_seq=3 ttl=64 time=0.725 ms

--- 2001:db8::10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.425/0.543/0.725/0.131 ms
```

Tùy chọn `-c` được sử dụng để chỉ định số lượng gói cần gửi. Nếu bỏ qua tùy chọn này, `ping` và `ping6` sẽ tiếp tục gửi các gói cho đến khi người dùng dừng nó (thường là bằng tổ hợp bàn phím `Ctrl + C`).

Chỉ vì không thể "ping" máy chủ không có nghĩa là chúng ta không thể kết nối với máy chủ đó. Nhiều tổ chức sẽ có tường lửa hoặc danh sách kiểm soát truy cập bộ định tuyến để chặn mọi thứ ngoại trừ mức tối thiểu cần thiết để hệ thống của họ hoạt động. Mức này cũng bao gồm yêu cầu và trả lời phản hồi ICMP. Vì các gói này có thể bao gồm dữ liệu tùy ý nên kẻ tấn công thông minh có thể sử dụng chúng để lấy cắp dữ liệu.

Truy vết Tuyến

Các chương trình `traceroute` và `traceroute6` có thể được sử dụng để hiển thị cho người dùng tuyến để một gói tin đi đến đích. Chúng thực hiện điều này bằng cách gửi nhiều gói đến đích và tăng trường *Time-To-Live* (TTL) của tiêu đề IP với mỗi gói tiếp theo. Mỗi bộ định tuyến trên đường đi sẽ phản hồi bằng thông báo ICMP vượt quá TTL:

```
$ traceroute 192.168.1.20
traceroute to 192.168.1.20 (192.168.1.20), 30 hops max, 60 byte packets
 1  10.0.2.2 (10.0.2.2)  0.396 ms  0.171 ms  0.132 ms
 2  192.168.1.20 (192.168.1.20)  2.665 ms  2.573 ms  2.573 ms

$ traceroute 192.168.50.2
traceroute to 192.168.50.2 (192.168.50.2), 30 hops max, 60 byte packets
```

```

1 192.168.50.2 (192.168.50.2) 0.433 ms 0.273 ms 0.171 ms
$ traceroute6 2001:db8::11
traceroute to 2001:db8::11 (2001:db8::11), 30 hops max, 80 byte packets
1 2001:db8::11 (2001:db8::11) 0.716 ms 0.550 ms 0.641 ms
$ traceroute 2001:db8::11
traceroute to 2001:db8::11 (2001:db8::11), 30 hops max, 80 byte packets
1 2001:db8::10 (2001:db8::11) 0.617 ms 0.461 ms 0.387 ms
$ traceroute net2.example.net
traceroute to net2.example.net (192.168.50.2), 30 hops max, 60 byte packets
1 net2.example.net (192.168.50.2) 0.533 ms 0.529 ms 0.504 ms
$ traceroute6 net2.example.net
traceroute to net2.example.net (2001:db8::11), 30 hops max, 80 byte packets
1 net2.example.net (2001:db8::11) 0.738 ms 0.607 ms 0.304 ms

```

Theo mặc định, `traceroute` sẽ gửi 3 gói UDP có dữ liệu rác đến cổng 33434 và tăng dần mỗi lần gửi một gói. Mỗi dòng trong đầu ra của lệnh là một giao diện bộ định tuyến mà gói đi qua. Thời gian hiển thị trên mỗi dòng của đầu ra là thời gian trễ trọn vòng cho mỗi gói. Địa chỉ IP là địa chỉ của giao diện bộ định tuyến được đề cập. Nếu có thể, `traceroute` sẽ sử dụng tên miền DNS của giao diện bộ định tuyến. Đôi khi chúng ta sẽ thấy `*` thay cho một mốc thời gian. Khi điều này xảy ra, điều đó có nghĩa là `traceroute` chưa bao giờ nhận được thông báo vượt quá TTL cho gói này. Khi chúng ta có thể nhìn thấy điều này thì thường có nghĩa là phản hồi cuối cùng là bước nhảy ngắn (hop) cuối cùng trên tuyến.

Nếu có quyền truy cập vào `root`, tùy chọn `-I` sẽ đặt `traceroute` để sử dụng các yêu cầu phản hồi ICMP thay vì các gói UDP. Điều này thường hiệu quả hơn UDP vì máy chủ đích có nhiều khả năng đáp ứng yêu cầu phản hồi ICMP hơn gói UDP:

```

# traceroute -I learning.lpi.org
traceroute to learning.lpi.org (208.94.166.201), 30 hops max, 60 byte packets
1 047-132-144-001.res.spectrum.com (47.132.144.1) 9.764 ms 9.702 ms 9.693 ms
2 096-034-094-106.biz.spectrum.com (96.34.94.106) 8.389 ms 8.481 ms 8.480 ms
3 dtr01h1rgnc-gbe-4-15.h1rg.nc.charter.com (96.34.64.172) 8.763 ms 8.775 ms 8.770 ms
4 acr01mgtnc-vln-492.mgtnc.nc.charter.com (96.34.67.202) 27.080 ms 27.154 ms 27.151 ms
5 bbr01gnv1sc-bue-3.gnv1.sc.charter.com (96.34.2.112) 31.339 ms 31.398 ms 31.395 ms
6 bbr01aldlmi-tge-0-0-0-13.aldl.mi.charter.com (96.34.0.161) 39.092 ms 38.794 ms 38.821
ms
7 prr01ashbva-bue-3.ashb.va.charter.com (96.34.3.51) 34.208 ms 36.474 ms 36.544 ms
8 bx2-ashburn.bell.ca (206.126.236.203) 53.973 ms 35.975 ms 38.250 ms
9 tcore4-ashburnbk_0-12-0-0.net.bell.ca (64.230.125.190) 66.315 ms 65.319 ms 65.345 ms
10 tcore4-toronto47_2-8-0-3.net.bell.ca (64.230.51.22) 67.427 ms 67.502 ms 67.498 ms
11 agg1-toronto47_xe-7-0-0_core.net.bell.ca (64.230.161.114) 61.270 ms 61.299 ms 61.291
ms

```

```

12 dis4-clarkson16_5-0.net.bell.ca (64.230.131.98) 61.101 ms 61.177 ms 61.168 ms
13 207.35.12.142 (207.35.12.142) 70.009 ms 70.069 ms 59.893 ms
14 unassigned-117.001.centrilogic.com (66.135.117.1) 61.778 ms 61.950 ms 63.041 ms
15 unassigned-116.122.akn.ca (66.135.116.122) 62.702 ms 62.759 ms 62.755 ms
16 208.94.166.201 (208.94.166.201) 62.936 ms 62.932 ms 62.921 ms

```

Một số tổ chức sẽ chặn yêu cầu và trả lời phản hồi ICMP. Để giải quyết vấn đề này, chúng ta có thể sử dụng TCP. Bằng cách sử dụng cổng TCP mở đã biết, ta có thể đảm bảo rằng máy chủ đích sẽ phản hồi. Để sử dụng TCP, hãy sử dụng tùy chọn `-T` cùng với `-p` để chỉ định cổng. Giống như các yêu cầu phản hồi ICMP, chúng ta phải có quyền truy cập vào `root` để thực hiện việc này:

```

# traceroute -m 60 -T -p 80 learning.lpi.org
traceroute to learning.lpi.org (208.94.166.201), 60 hops max, 60 byte packets
 1 * * *
 2 096-034-094-106.biz.spectrum.com (96.34.94.106) 12.178 ms 12.229 ms 12.175 ms
 3 dtr01h1rgnc-gbe-4-15.h1rg.nc.charter.com (96.34.64.172) 12.134 ms 12.093 ms 12.062 ms
 4 acr01mgtnc-vln-492.mgtnc.nc.charter.com (96.34.67.202) 31.146 ms 31.192 ms 31.828 ms
 5 bbr01gnv1sc-bue-3.gnv1.sc.charter.com (96.34.2.112) 39.057 ms 46.706 ms 39.745 ms
 6 bbr01aldlmi-tge-0-0-0-13.aldl.mi.charter.com (96.34.0.161) 50.590 ms 58.852 ms 58.841
ms
 7 prr01ashbva-bue-3.ashb.va.charter.com (96.34.3.51) 34.556 ms 37.892 ms 38.274 ms
 8 bx2-ashburn.bell.ca (206.126.236.203) 38.249 ms 36.991 ms 36.270 ms
 9 tcore4-ashburnbk_0-12-0-0.net.bell.ca (64.230.125.190) 66.779 ms 63.218 ms tcore3-
ashburnbk_100ge0-12-0-0.net.bell.ca (64.230.125.188) 60.441 ms
10 tcore4-toronto47_2-8-0-3.net.bell.ca (64.230.51.22) 63.932 ms 63.733 ms 68.847 ms
11 agg2-toronto47_xe-7-0-0_core.net.bell.ca (64.230.161.118) 60.144 ms 60.443 ms agg1-
toronto47_xe-7-0-0_core.net.bell.ca (64.230.161.114) 60.851 ms
12 dis4-clarkson16_5-0.net.bell.ca (64.230.131.98) 67.246 ms dis4-clarkson16_7-
0.net.bell.ca (64.230.131.102) 68.404 ms dis4-clarkson16_5-0.net.bell.ca (64.230.131.98)
67.403 ms
13 207.35.12.142 (207.35.12.142) 66.138 ms 60.608 ms 64.656 ms
14 unassigned-117.001.centrilogic.com (66.135.117.1) 70.690 ms 62.190 ms 61.787 ms
15 unassigned-116.122.akn.ca (66.135.116.122) 62.692 ms 69.470 ms 68.815 ms
16 208.94.166.201 (208.94.166.201) 61.433 ms 65.421 ms 65.247 ms
17 208.94.166.201 (208.94.166.201) 64.023 ms 62.181 ms 61.899 ms

```

Giống như `ping`, `traceroute` cũng có những hạn chế. Tường lửa và bộ định tuyến có thể chặn các gói được gửi từ hoặc gửi trả `traceroute`. Nếu có quyền truy cập `root`, có những tùy chọn có thể giúp người dùng nhận được các kết quả chính xác.

Tìm MTU bằng tracepath

Lệnh `tracepath` cũng tương tự như `traceroute`. Sự khác biệt là nó sẽ theo dõi các kích thước Đơn vị truyền tối đa (MTU) dọc theo đường dẫn. MTU là cài đặt được định cấu hình trên giao diện mạng hoặc giới hạn phần cứng của đơn vị dữ liệu giao thức lớn nhất mà nó có thể truyền hoặc nhận. Chương trình `tracepath` hoạt động theo một cách tương tự như `traceroute` ở chỗ nó sẽ tăng TTL với mỗi gói và khác ở chỗ sẽ gửi một gói dữ liệu UDP rất lớn. Hầu như không thể tránh khỏi việc gói dữ liệu phải lớn hơn thiết bị có MTU nhỏ nhất dọc theo tuyến. Khi gói đến thiết bị này, thiết bị thường sẽ phản hồi bằng một gói không thể truy cập đích. Gói không thể truy cập đích ICMP có một trường dành cho MTU của liên kết mà nó sẽ gửi gói đi nếu có thể. `tracepath` sau đó sẽ gửi tất cả các gói tiếp theo có kích thước này:

```
$ tracepath 192.168.1.20
1?: [LOCALHOST] pmtu 1500
1: 10.0.2.2 0.321ms
1: 10.0.2.2 0.110ms
2: 192.168.1.20 2.714ms reached
Resume: pmtu 1500 hops 2 back 64
```

Không giống như `traceroute`, chúng ta phải đặc biệt sử dụng `tracepath6` cho IPv6:

```
$ tracepath 2001:db8::11
tracepath: 2001:db8::11: Address family for hostname not supported
$ tracepath6 2001:db8::11
1?: [LOCALHOST] 0.027ms pmtu 1500
1: net2.example.net 0.917ms reached
1: net2.example.net 0.527ms reached
Resume: pmtu 1500 hops 1 back 1
```

Đầu ra cũng tương tự như `traceroute`. Ưu điểm của `tracepath` là ở dòng cuối cùng: nó xuất ra MTU nhỏ nhất trên toàn bộ liên kết. Điều này có thể hữu ích cho việc khắc phục sự cố các kết nối không thể xử lý các đoạn nhỏ.

Giống như các công cụ khắc phục sự cố trước đây, thiết bị cũng có khả năng chặn các gói.

Tạo Kết nối tùy ý

Chương trình `nc` (tức `netcat`) có thể gửi hoặc nhận dữ liệu tùy ý qua kết nối mạng TCP hoặc UDP. Các ví dụ sau đây sẽ làm rõ chức năng của nó.

Dưới đây là ví dụ về thiết lập một trình nghe trên cổng 1234:

```
$ nc -l 1234
LPI Example
```

Đầu ra của `LPI Example` sẽ xuất hiện sau ví dụ bên dưới. Ví dụ này đang thiết lập một người gửi netcat để gửi các gói đến `net2.example.net` trên cổng 1234. Tùy chọn `-l` được sử dụng để chỉ định rằng người dùng muốn `nc` nhận dữ liệu thay vì gửi nó đi:

```
$ nc net2.example.net 1234
LPI Example
```

Nhấn `Ctrl` + `C` trên một trong hai hệ thống để dừng kết nối.

Netcat có thể làm việc với cả địa chỉ IPv4 và IPv6. Nó cũng hoạt động với cả TCP và UDP. Nó thậm chí có thể được sử dụng để thiết lập một vỏ từ xa.

WARNING

Hãy lưu ý rằng không phải mọi cài đặt `nc` đều có hỗ trợ khóa chuyển `-e`. Hãy nhớ xem lại các trang hướng dẫn để biết thông tin bảo mật về tùy chọn này cũng như các phương pháp thay thế để thực thi lệnh trên hệ thống từ xa.

```
$ hostname
net2
$ nc -u -e /bin/bash -l 1234
```

Tùy chọn `-u` là để dành cho UDP. `-e` sẽ hướng dẫn netcat gửi mọi thứ nó nhận được đến đầu vào tiêu chuẩn của tệp thực thi theo sau nó. Trong ví dụ này là `/bin/bash`.

```
$ hostname
net1
$ nc -u net2.example.net 1234
hostname
net2
pwd
/home/emma
```

Ta có thể thấy đầu ra lệnh `hostname` khớp với đầu ra của máy chủ đang nghe và lệnh `pwd` đã xuất ra một thư mục.

Xem các Kết nối và Trình Nghe hiện tại

Các chương trình `netstat` và `ss` có thể được sử dụng để xem trạng thái của các trình nghe và kết nối hiện tại của người dùng. Giống như `ifconfig`, `netstat` là một công cụ kế thừa. Cả `netstat` và `ss` đều có đầu ra và tùy chọn tương tự như nhau. Dưới đây là một số tùy chọn có sẵn cho cả hai chương trình:

-a

Hiển thị tất cả các ổ nối.

-l

Hiển thị các ổ nối đang nghe.

-p

Hiển thị các tiến trình được liên kết với kết nối.

-n

Ngăn chặn việc tra cứu tên cho cả cổng và địa chỉ.

-t

Hiển thị các kết nối TCP.

-u

Hiển thị các kết nối UDP.

Các ví dụ bên dưới cho thấy đầu ra của một bộ tùy chọn thường được sử dụng cho cả hai chương trình:

```
# netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program
name
tcp      0      0 0.0.0.0:22              0.0.0.0:*               LISTEN     892/sshd
tcp      0      0 127.0.0.1:25            0.0.0.0:*               LISTEN     1141/master
tcp6     0      0 :::22                  :::*                   LISTEN     892/sshd
tcp6     0      0 :::1:25                 :::*                   LISTEN     1141/master
udp      0      0 0.0.0.0:68             0.0.0.0:*               692/dhclient

# ss -tulnp
# ss -tulnp
Netid  State      Recv-Q Send-Q   Local Address:Port      Peer
Address:Port
udp    UNCONN    0      0           *:68                    *:
```

```

users: ("dhclient", pid=693, fd=6)
tcp  LISTEN  0  128  *:*
users: ("sshd", pid=892, fd=3)
tcp  LISTEN  0  100  127.0.0.1:25
users: ("master", pid=1099, fd=13)
tcp  LISTEN  0  128  [::]:22  [::]:*
users: ("sshd", pid=892, fd=4)
tcp  LISTEN  0  100  [::1]:25  [::]:*
users: ("master", pid=1099, fd=14)

```

Cột `Recv-Q` là số gói mà một ổ nối đã nhận được nhưng không được chuyển đến chương trình của nó. Cột `Send-Q` là số gói mà một ổ nối đã gửi mà chưa được người nhận xác nhận. Các cột còn lại đã tự giải thích chính bản thân chúng.

Bài tập Hướng dẫn

1. Bạn sẽ sử dụng (những) lệnh nào để gửi phản hồi ICMP tới `learning.lpi.org`?

2. Làm thế nào để có thể xác định được tuyến tới `8.8.8.8`?

3. Lệnh nào sẽ hiển thị cho người dùng biết nếu có bất kỳ tiến trình nào đang nghe trên cổng TCP 80?

4. Làm thế nào để có thể tìm thấy tiến trình nào đang nghe trên một cổng?

5. Làm cách nào để có thể xác định MTU tối đa của một đường dẫn mạng?

Bài tập Mở rộng

1. Làm cách nào để có thể sử dụng netcat để gửi yêu cầu HTTP đến máy chủ web?

2. Một số lý do khiến việc "ping" máy chủ có thể không thành công là gì?

3. Hãy kể tên một công cụ mà bạn có thể sử dụng để xem các gói mạng đến hoặc rời khỏi máy chủ Linux?

4. Làm cách nào để có thể buộc traceroute sử dụng một giao diện khác?

5. Traceroute có thể báo cáo các MTU không?

Tóm tắt

Kết nối mạng thường được cấu hình bởi các tệp lệnh khởi động của hệ thống hoặc một trình trợ giúp như NetworkManager. Hầu hết các bản phân phối đều có các công cụ giúp người dùng chỉnh sửa các tệp cấu hình tệp lệnh khởi động. Hãy tham khảo tài liệu của bản phân phối để biết thêm chi tiết.

Khả năng cấu hình mạng theo cách thủ công cho phép chúng ta khắc phục sự cố hiệu quả hơn. Nó rất hữu ích trong các môi trường tối thiểu được sử dụng cho những việc như khôi phục từ bản sao lưu hoặc di chuyển sang phần cứng mới.

Các tiện ích được nhắc tới trong bài học này có nhiều chức năng hơn so với những gì đã được đề cập. Việc xem trang hướng dẫn của từng chức năng để làm quen với các tùy chọn có sẵn sẽ rất hữu ích đối với người dùng. Các lệnh `ss` và `ip` là cách thực hiện hiện đại, trong khi những lệnh còn lại đã được đề cập đến được coi là các công cụ cũ (tuy vẫn được sử dụng khá phổ biến).

Cách tốt nhất để làm quen với các công cụ được đề cập tới là thực hành chúng. Chỉ với một máy tính có dung lượng RAM khiêm tốn, chúng ta đã có thể thiết lập một phòng thí nghiệm mạng ảo bằng các máy ảo để thực hành. Chỉ cần tới ba máy ảo là đã đủ để chúng ta làm quen với các công cụ đã được nhắc tới.

Các lệnh được sử dụng trong bài học này bao gồm:

ping và **ping6**

Được sử dụng để truyền các gói ICMP đến máy chủ từ xa để kiểm tra tính khả dụng của kết nối mạng.

traceroute và **traceroute6**

Được sử dụng để truy vết đường dẫn qua mạng nhằm xác định khả năng kết nối của mạng.

tracepath và **tracepath6**

Được sử dụng để truy vết đường dẫn qua mạng cũng như xác định kích thước MTU dọc theo tuyến.

nc

Được sử dụng để thiết lập các kết nối tùy ý trên mạng nhằm kiểm tra kết nối cũng như truy vấn mạng về các dịch vụ và thiết bị có sẵn.

netstat

Lệnh kế thừa được sử dụng để xác định số liệu thống kê và kết nối mạng mở của hệ thống.

SS

Lệnh hiện đại được sử dụng để xác định số liệu thống kê và kết nối mạng mở của hệ thống.

Đáp án Bài tập Hướng dẫn

1. Bạn sẽ sử dụng (những) lệnh nào để gửi tiếng vang ICMP tới `learning.lpi.org`?

Lệnh `ping` hoặc `ping6`:

```
$ ping learning.lpi.org
```

hoặc

```
$ ping6 learning.lpi.org
```

2. Làm thế nào để có thể xác định được tuyến tới `8.8.8.8`?

Bằng cách sử dụng lệnh `tracpath` hoặc `tracert`.

```
$ tracpath 8.8.8.8
```

hoặc

```
$ traceroute 8.8.8.8
```

3. Lệnh nào sẽ hiển thị cho người dùng biết nếu có bất kỳ tiến trình nào đang nghe trên cổng TCP 80?

Với `ss`:

```
$ ss -ln | grep ":80"
```

Với `netstat`:

```
$ netstat -ln | grep ":80"
```

Mặc dù không được coi là yêu cầu của bài kiểm tra nhưng bạn cũng có thể sử dụng `lsof`:

```
# lsof -Pi:80
```


4. Làm thế nào để có thể tìm thấy tiến trình nào đang nghe trên một cổng?

Một lần nữa, có nhiều cách để thực hiện việc này. Bạn có thể sử dụng `lsof` theo cách tương tự như câu trả lời trước để thay thế số cổng. Bạn cũng có thể sử dụng `netstat` hoặc `ss` với tùy chọn `-p`. Hãy nhớ rằng, `netstat` được coi là một công cụ kế thừa.

```
# netstat -lnp | grep ":22"
```

Các tùy chọn tương tự khả dụng với `netstat` cũng khả dụng với `ss`:

```
# ss -lnp | grep ":22"
```

5. Làm cách nào để có thể xác định MTU tối đa của một đường dẫn mạng?

Bằng cách sử dụng lệnh `tracert`:

```
$ tracert somehost.example.com
```

Đáp án Bài tập Mở rộng

1. Làm cách nào để có thể sử dụng netcat để gửi yêu cầu HTTP đến máy chủ web?

Bằng cách nhập dòng yêu cầu HTTP, bất kỳ một tiêu đề nào và một dòng trống vào cửa sổ dòng lệnh:

```
$ nc learning.lpi.org 80
GET /index.html HTTP/1.1
HOST: learning.lpi.org

HTTP/1.1 302 Found
Location: https://learning.lpi.org:443/index.html
Date: Wed, 27 May 2020 22:54:46 GMT
Content-Length: 5
Content-Type: text/plain; charset=utf-8

Found
```

2. Một số lý do khiến việc ping máy chủ có thể không thành công là gì?

Có một số lý do có thể gây ra việc này:

- Máy chủ từ xa không hoạt động.
- Một ACL bộ định tuyến đang chặn ping của bạn.
- Tường lửa của máy chủ từ xa đang chặn ping của bạn.
- Bạn có thể đang sử dụng tên hoặc địa chỉ máy chủ không chính xác.
- Việc phân giải tên miền của bạn đang trả về một địa chỉ không chính xác.
- Cấu hình mạng của máy không đúng.
- Tường lửa của máy đang chặn nó.
- Cấu hình mạng của máy chủ từ xa không chính xác.
- Các giao diện trên máy bị ngắt kết nối.
- Các giao diện của máy từ xa bị ngắt kết nối.
- Một thành phần mạng như bộ chuyển mạch, cáp hoặc bộ định tuyến giữa máy của bạn và máy điều khiển từ xa không còn hoạt động.

3. Hãy kể tên một công cụ mà bạn có thể sử dụng để xem các gói mạng đến hoặc rời khỏi máy chủ Linux?

Cả `tcpdump` và `wireshark` đều có thể được sử dụng.

4. Làm cách nào để có thể buộc `traceroute` sử dụng một giao diện khác?

Bằng cách sử dụng tùy chọn `-i`:

```
$ traceroute -i eth2 learning.lpi.org
traceroute -i eth2 learning.lpi.org
traceroute to learning.lpi.org (208.94.166.201), 30 hops max, 60 byte packets
...
```

5. `Traceroute` có thể báo cáo các MTU không?

Có thể, với tùy chọn `--mtu`:

```
# traceroute -I --mtu learning.lpi.org
traceroute to learning.lpi.org (208.94.166.201), 30 hops max, 65000 byte packets
 1 047-132-144-001.res.spectrum.com (47.132.144.1) 9.974 ms F=1500 10.476 ms 4.743 ms
 2 096-034-094-106.biz.spectrum.com (96.34.94.106) 8.697 ms 9.963 ms 10.321 ms
...
```



109.4 Định cấu hình DNS phía Máy Khách

Tham khảo các mục tiêu LPI

[LPIC-1 version 5.0, Exam 102, Objective 109.4](#)

Khối lượng

2

Các lĩnh vực kiến thức chính

- Truy vấn máy chủ DNS từ xa.
- Định cấu hình độ phân giải tên cục bộ và sử dụng máy chủ DNS từ xa.
- Sửa đổi thứ tự thực hiện phân giải tên.
- Gỡ lỗi liên quan đến việc phân giải tên.
- Kiến thức về systemd-resolved

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- `/etc/hosts`
- `/etc/resolv.conf`
- `/etc/nsswitch.conf`
- `host`
- `dig`
- `getent`



109.4 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	109 Những yếu tố cơ bản về Mạng
Mục tiêu:	109.4 Định cấu hình DNS phía Máy Khách
Bài:	1 trên 1

Giới thiệu

Bài học này sẽ thảo luận về việc định cấu hình phân giải tên miền phía máy khách và cách sử dụng một số công cụ phân giải tên miền giao diện dòng lệnh CLI.

Để ghi nhớ và duy trì địa chỉ IP, UID và GID cũng như các chỉ số khác cho tất cả mọi thứ không phải là một việc khả thi. Các dịch vụ phân giải tên miền sẽ dịch các tên miền dễ nhớ sang thành số và ngược lại. Bài học này sẽ tập trung vào việc phân giải tên miền máy chủ, nhưng một tiến trình tương tự cũng sẽ xảy ra với tên miền người dùng, nhóm, số cổng và các tên khác.

Tiến trình phân giải Tên miền

Các chương trình phân giải tên miền thành số hầu như luôn sử dụng các hàm được cung cấp bởi thư viện C tiêu chuẩn. Nó cũng chính là glibc của dự án GNU trên hệ thống Linux. Điều đầu tiên các hàm này sẽ làm là đọc tệp `/etc/nsswitch.conf` để biết hướng dẫn về cách phân giải loại tên miền đó. Bài học này sẽ tập trung vào việc phân giải tên máy chủ, nhưng một tiến trình tương tự cũng sẽ được áp dụng cho các kiểu phân giải tên miền khác. Sau khi tiến trình đọc `/etc/nsswitch.conf`, hàm sẽ tra cứu tên miền theo cách được chỉ định. Vì `/etc/nsswitch.conf` có hỗ trợ các tiện ích bổ sung nên ở bước tiếp theo chúng ta có thể thực

hiện bất kỳ một tác vụ nào. Sau khi hàm tra cứu tên miền hoặc số xong, nó sẽ trả kết quả cho tiến trình gọi.

Các Hạng DNS

DNS có ba hạng bản ghi là IN, HS và CH. Trong bài học này, tất cả các truy vấn DNS sẽ thuộc loại IN. Hạng IN được dành cho các địa chỉ internet sử dụng chồng TCP/IP. CH được dành cho ChaosNet - một công nghệ mạng đã tồn tại trong một khoảng thời gian ngắn và không còn được sử dụng nữa. Hạng HS dành cho Hesiod. Hesiod là một cách để lưu trữ những thứ như mật khẩu và các mục nhập nhóm trong DNS. Hesiod nằm ngoài phạm vi của bài học này.

Hiểu về `/etc/nsswitch.conf`

Cách tốt nhất để tìm hiểu về tệp này là đọc trang hướng dẫn (cũng chính là một phần của dự án trang hướng dẫn Linux). Nó có sẵn trên hầu hết các hệ thống và có thể được truy cập bằng lệnh `man nsswitch.conf`. Ngoài ra, nó cũng có thể được tìm thấy tại https://man7.org/linux/man-pages/dir_section_5.html

Dưới đây là một ví dụ đơn giản về `/etc/nsswitch.conf` từ trang hướng dẫn của nó:

```
passwd:          compat
group:           compat
shadow:         compat

hosts:           dns [!UNAVAIL=return] files
networks:        nis [!NOTFOUND=return] files
ethers:          nis [!NOTFOUND=return] files
protocols:       nis [!NOTFOUND=return] files
rpc:             nis [!NOTFOUND=return] files
services:        nis [!NOTFOUND=return] files
# This is a comment. It is ignored by the resolution functions.
```

Tệp đã được tổ chức thành các cột. Cột ngoài cùng bên trái là loại của cơ sở dữ liệu tên miền. Các cột còn lại là các phương thức mà hàm phân giải nên sử dụng để tra cứu tên miền. Tiếp theo sau các phương thức là các hàm từ trái sang phải. Các cột có ký hiệu `[]` sẽ được sử dụng để cung cấp một số logic có điều kiện hạn chế cho cột nằm ở bên trái nó.

Giả sử một tiến trình đang cố gắng phân giải tên máy chủ `learning.lpi.org`. Nó sẽ thực hiện một lệnh gọi thư viện C thích hợp (rất có thể là `gethostbyname`). Hàm này sau đó sẽ đọc `/etc/nsswitch.conf`. Vì tiến trình đang tra cứu một tên máy chủ nên nó sẽ tìm dòng được bắt đầu bằng `hosts`. Sau đó, nó sẽ cố gắng sử dụng DNS để phân giải tên miền. Cột

[!UNAVAIL=return] tiếp theo có nghĩa là nếu dịch vụ *không phải* là không khả dụng thì đừng thử nguồn tiếp theo - tức là nếu DNS khả dụng, hãy ngừng cố gắng phân giải tên máy chủ ngay cả khi máy chủ định danh không thể làm được. Nếu DNS không khả dụng thì hãy tiếp tục với nguồn tiếp theo. Trong trường hợp này, nguồn tiếp theo sẽ là `file`.

Khi một cột ở định dạng [result=action] thì có nghĩa là khi việc tra cứu trình phân giải được thực hiện ở cột bên trái của cột điều kiện logic là `result` thì `action` sẽ được thực hiện. Nếu trước `result` có `!` thì có nghĩa là nếu kết quả không phải là `result`, hãy thực hiện `action`. Để biết mô tả về các kết quả và hành động có thể xảy ra, hãy xem trang hướng dẫn.

Bây giờ, giả sử một tiến trình đang cố gắng phân giải một số (của) cổng thành tên miền dịch vụ, nó sẽ đọc dòng `services`. Nguồn đầu tiên được liệt kê sẽ là `NIS`. `NIS` là viết tắt của *Network Information Service* (Dịch vụ Thông tin Mạng; đôi khi nó được gọi là các trang vàng - *yellow pages*). Đây là một dịch vụ cũ cho phép quản lý tập trung nhiều thứ (ví dụ như người dùng). Nó hiếm khi còn được sử dụng do tính bảo mật kém. Cột [NOTFOUND=return] tiếp theo có nghĩa là nếu tra cứu thành công nhưng không tìm thấy dịch vụ thì hãy dừng tìm kiếm. Nếu điều kiện nói trên không áp dụng được, hãy sử dụng các tệp cục bộ.

Bất cứ điều gì ở bên phải ký hiệu `#` đều là một chú thích và sẽ bị bỏ qua.

Tệp `/etc/resolv.conf`

Tệp `/etc/resolv.conf` được sử dụng để định cấu hình việc phân giải máy chủ qua DNS. Một số bản phân phối sẽ có tệp lệnh khởi động, trình nền và các công cụ khác ghi vào tệp này. Hãy ghi nhớ điều này khi chỉnh sửa thủ công tệp này và kiểm tra bản phân phối của mình và mọi tài liệu về công cụ cấu hình mạng nếu cần chỉnh sửa thủ công. Một số công cụ như `NetworkManager` sẽ để lại một chú thích trong tệp để cho người dùng biết rằng các thay đổi thủ công sẽ bị ghi đè.

Đối với `/etc/nsswitch.conf`, có một trang hướng dẫn được liên kết với tệp. Nó có thể được truy cập bằng lệnh `man resolv.conf` hoặc tại <https://man7.org/linux/man-pages/man5/resolv.conf.5.html>.

Định dạng của tệp cũng khá dễ hiểu. Ở cột ngoài cùng bên trái, chúng ta có tùy chọn `name`. Các cột còn lại trên cùng một dòng là giá trị của tùy chọn.

Tùy chọn phổ biến nhất là tùy chọn `nameserver`. Nó được sử dụng để chỉ định địa chỉ IPv4 hoặc IPv6 của máy chủ DNS. Tính đến ngày cuốn sách này được viết, chúng ta có thể chỉ định tối đa ba máy chủ tên miền. Nếu `/etc/resolv.conf` không có tùy chọn `nameserver`, hệ thống của người dùng theo mặc định sẽ sử dụng máy chủ tên miền trên máy cục bộ.

Dưới đây là một tệp ví dụ đơn giản đại diện cho các cấu hình phổ biến:

```
search lpi.org
nameserver 10.0.0.53
nameserver fd00:ffff::2:53
```

Tùy chọn `search` được sử dụng để cho phép người dùng tìm kiếm dạng ngắn. Trong ví dụ này, một miền tìm kiếm `lpi.org` đã được định cấu hình. Điều này có nghĩa là mọi nỗ lực phân giải tên máy chủ mà không có phần tên miền sẽ được thêm `.lpi.org` vào trước cụm tìm kiếm. Ví dụ: nếu ta cố gắng tìm kiếm máy chủ có tên `learning`, trình phân giải sẽ tìm kiếm `learning.lpi.org`. Chúng ta có thể cấu hình tối đa sáu miền tìm kiếm.

Một tùy chọn phổ biến khác là tùy chọn `domain`. Tùy chọn này được sử dụng để đặt tên miền cục bộ của người dùng. Nếu bị thiếu, tùy chọn này sẽ mặc định là mọi thứ đứng sau ký tự `.` đầu tiên trong tên máy chủ của máy. Nếu tên máy chủ không có chứa `.` thì nó sẽ giả định rằng máy là một phần của miền gốc. Giống như `search`, `domain` cũng có thể được sử dụng để tìm kiếm các tên miền ngắn.

Hãy nhớ rằng `domain` và `search` sẽ loại trừ lẫn nhau. Nếu cả hai đều có mặt, cái xuất hiện cuối cùng trong tệp sẽ được sử dụng.

Có một số tùy chọn có thể được đặt để can thiệp vào hoạt động của trình phân giải. Để đặt những tùy chọn này, hãy sử dụng từ khóa `options`, theo sau là tên của tùy chọn cần đặt và nếu có thể, hãy thêm giá trị vào sau `:`. Dưới đây là một ví dụ về cách đặt tùy chọn thời gian chờ - tức khoảng thời gian tính bằng giây mà trình phân giải sẽ đợi một máy chủ tên miền trước khi ngừng lại:

```
option timeout:3
```

`resolv.conf` cũng có các tùy chọn khác nữa, nhưng trên đây là những tùy chọn phổ biến nhất.

Tệp `/etc/hosts`

Tệp `/etc/hosts` được sử dụng để phân giải tên thành địa chỉ IP và ngược lại. Cả IPv4 và IPv6 đều được hỗ trợ. Cột bên trái là địa chỉ IP, còn lại là các tên miền gắn liền với địa chỉ đó. Cách sử dụng phổ biến nhất cho `/etc/hosts` là dành cho các máy chủ và địa chỉ không thể sử dụng DNS (chẳng hạn như các địa chỉ lặp vòng). Trong ví dụ bên dưới, địa chỉ IP của các thành phần cơ sở hạ tầng quan trọng đã được xác định.

Sau đây là một ví dụ thực tế về tệp `/etc/hosts`:

```
127.0.0.1    localhost
127.0.1.1    proxy
```



```

::1          localhost ip6-localhost ip6-loopback
ff02::1     ip6-allnodes
ff02::2     ip6-allrouters

10.0.0.1    gateway.lpi.org gateway gw
fd00:ffff::1 gateway.lpi.org gateway gw

10.0.1.53   dns1.lpi.org
fd00:ffff::1:53 dns1.lpi.org
10.0.2.53   dns2.lpi.org
fd00:ffff::2:53 dns2.lpi.org

```

systemd-resolved

Systemd cung cấp một dịch vụ có tên là `systemd-resolved`. Nó sẽ cung cấp mDNS, DNS và LLMNR. Khi chạy, nó sẽ lắng nghe các yêu cầu DNS trên `127.0.0.53`. Nó *không* cung cấp một máy chủ DNS chính thức. Mọi yêu cầu DNS mà nó nhận được đều sẽ được tra cứu bằng cách truy vấn các máy chủ được định cấu hình trong `/etc/systemd/resolv.conf` hoặc `/etc/resolv.conf`. Nếu muốn sử dụng dịch vụ này, hãy sử dụng `resolve` cho `hosts` trong `/etc/nsswitch.conf`. Hãy nhớ rằng gói Hệ điều hành sở hữu thư viện `systemd-resolved` có thể sẽ không được cài đặt theo mặc định.

Công cụ phân giải Tên miền

Có nhiều công cụ có sẵn dành cho người dùng Linux dùng để phân giải tên. Bài học này sẽ đề cập tới ba công cụ: `getent` (rất hữu ích trong việc xem xét các yêu cầu trong thế giới thực sẽ được phân giải như thế nào), `host` (rất hữu ích cho các truy vấn DNS đơn giản) và `dig` (rất hữu ích cho các hoạt động DNS phức tạp có thể hỗ trợ khắc phục sự cố máy chủ DNS).

Lệnh `getent`

Tiện ích `getent` được sử dụng để hiển thị các mục từ cơ sở dữ liệu dịch vụ tên miền. Nó có thể truy xuất các bản ghi từ bất kỳ một nguồn nào có thể định cấu hình bằng `/etc/nsswitch.conf`.

Để sử dụng `getent`, chúng ta chỉ cần thêm loại tên miền cần phân giải sau nó và một mục nhập cụ thể tùy chọn để tra cứu. Nếu chỉ được chỉ định loại tên miền, `getent` sẽ cố gắng hiển thị tất cả các mục của loại dữ liệu đó:

```

$ getent hosts
127.0.0.1    localhost
127.0.1.1    proxy

```

```
10.0.1.53      dns1.lpi.org
10.0.2.53      dns2.lpi.org
127.0.0.1      localhost ip6-localhost ip6-loopback
$ getent hosts dns1.lpi.org
fd00:ffff::1:53 dns1.lpi.org
```

Bắt đầu với glibc phiên bản 2.2.5, chúng ta có thể buộc `getent` sử dụng một nguồn dữ liệu cụ thể bằng tùy chọn `-s`. Ví dụ dưới đây sẽ chứng minh điều này:

```
$ getent -s files hosts learning.lpi.org
::1          learning.lpi.org
$ getent -s dns hosts learning.lpi.org
208.94.166.198 learning.lpi.org
```

Lệnh `host`

`host` là một chương trình đơn giản để tra cứu các mục DNS. Nếu không có tùy chọn nào và `host` được cung cấp một tên miền, nó sẽ trả về các bộ bản ghi A, AAAA và MX. Nếu được cung cấp địa chỉ IPv4 hoặc IPv6, nó sẽ xuất ra bản ghi PTR nếu có sẵn:

```
$ host wikipedia.org
wikipedia.org has address 208.80.154.224
wikipedia.org has IPv6 address 2620:0:861:ed1a::1
wikipedia.org mail is handled by 10 mx1001.wikimedia.org.
wikipedia.org mail is handled by 50 mx2001.wikimedia.org.
$ host 208.80.154.224
224.154.80.208.in-addr.arpa domain name pointer text-lb.eqiad.wikimedia.org.
```

Nếu cần tìm kiếm một loại bản ghi cụ thể, chúng ta có thể sử dụng `host -t`:

```
$ host -t NS lpi.org
lpi.org name server dns1.easydns.com.
lpi.org name server dns3.easydns.ca.
lpi.org name server dns2.easydns.net.
$ host -t SOA lpi.org
lpi.org has SOA record dns1.easydns.com. zone.easydns.com. 1593109612 3600 600 1209600 300
```

`host` cũng có thể được sử dụng để truy vấn một máy chủ định danh cụ thể nếu chúng ta không muốn sử dụng các máy chủ đó trong `/etc/resolv.conf`. Ta chỉ cần thêm địa chỉ IP hoặc tên máy chủ của máy chủ cần sử dụng làm đối số cuối cùng:

```
$ host -t MX lpi.org dns1.easydns.com
Using domain server:
Name: dns1.easydns.com
Address: 64.68.192.10#53
Aliases:

lpi.org mail is handled by 10 aspmx4.googlemail.com.
lpi.org mail is handled by 10 aspmx2.googlemail.com.
lpi.org mail is handled by 5 alt1.aspmx.l.google.com.
lpi.org mail is handled by 0 aspmx.l.google.com.
lpi.org mail is handled by 10 aspmx5.googlemail.com.
lpi.org mail is handled by 10 aspmx3.googlemail.com.
lpi.org mail is handled by 5 alt2.aspmx.l.google.com.
```

Lệnh dig

Một công cụ khác để truy vấn máy chủ DNS là `dig`. Lệnh này chi tiết hơn rất nhiều so với `host`. Theo mặc định, `dig` sẽ truy vấn cho bản ghi A. Dù chỉ đơn giản là tra cứu một địa chỉ IP hoặc tên máy chủ nhưng nó vẫn sẽ khá dài dòng. `dig` cũng có thể được sử dụng để tra cứu đơn giản, nhưng nó sẽ phù hợp hơn trong việc khắc phục sự cố cấu hình máy chủ DNS:

```
$ dig learning.lpi.org

; <<>> DiG 9.11.5-P4-5.1+deb10u1-Debian <<>> learning.lpi.org
;; global options: +cmd
;; Got answer:
;; ->HEADER<<- opcode: QUERY, status: NOERROR, id: 63004
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 5

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: ca7a415be1cec45592b082665ef87f3483b81ddd61063c30 (good)
;; QUESTION SECTION:
;learning.lpi.org.      IN  A

;; ANSWER SECTION:
learning.lpi.org.     600 IN  A    208.94.166.198

;; AUTHORITY SECTION:
lpi.org.              86400 IN  NS   dns2.easydns.net.
lpi.org.              86400 IN  NS   dns1.easydns.com.
lpi.org.              86400 IN  NS   dns3.easydns.ca.
```

```
;; ADDITIONAL SECTION:
dns1.easydns.com. 172682 IN A 64.68.192.10
dns2.easydns.net. 170226 IN A 198.41.222.254
dns1.easydns.com. 172682 IN AAAA 2400:cb00:2049:1::a29f:1835
dns2.easydns.net. 170226 IN AAAA 2400:cb00:2049:1::c629:defe

;; Query time: 135 msec
;; SERVER: 192.168.1.20#53(192.168.1.20)
;; WHEN: Sun Jun 28 07:29:56 EDT 2020
;; MSG SIZE rcvd: 266
```

Như có thể thấy, `dig` cung cấp rất nhiều thông tin với đầu ra được phân chia thành các phần. Phần đầu tiên sẽ hiển thị thông tin về phiên bản `dig` được cài đặt và truy vấn đã gửi cùng với bất kỳ một tùy chọn nào được sử dụng cho lệnh. Tiếp theo, nó sẽ hiển thị thông tin về truy vấn và phản hồi.

Phần tiếp theo sẽ hiển thị thông tin về các phần mở rộng EDNS được sử dụng và truy vấn. Trong ví dụ này, phần mở rộng cookie sẽ được sử dụng. `dig` đang tìm bản ghi A cho `learning.lpi.org`.

Phần tiếp theo sẽ hiển thị kết quả của truy vấn. Số ở cột thứ hai chính là TTL của tài nguyên được tính bằng giây.

Phần còn lại của đầu ra sẽ cung cấp thông tin về các máy chủ tên miền của miền bao gồm các bản ghi NS cho máy chủ cùng với các bản ghi A và AAAA của các máy chủ trong bản ghi NS của miền.

Giống như `host`, chúng ta cũng có thể chỉ định loại bản ghi bằng tùy chọn `-t`:

```
$ dig -t SOA lpi.org

; <<>> DiG 9.11.5-P4-5.1+deb10u1-Debian <<>> -t SOA lpi.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16695
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 6

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 185c67140a63baf46c4493215ef8906f7bfbe15bdca3b01a (good)
;; QUESTION SECTION:
;lpi.org.          IN SOA

;; ANSWER SECTION:
```

```

lpi.org.      600 IN  SOA dns1.easydns.com. zone.easydns.com. 1593109612 3600 600 1209600
300

;; AUTHORITY SECTION:
lpi.org.      81989 IN  NS  dns1.easydns.com.
lpi.org.      81989 IN  NS  dns2.easydns.net.
lpi.org.      81989 IN  NS  dns3.easydns.ca.

;; ADDITIONAL SECTION:
dns1.easydns.com. 168271 IN  A   64.68.192.10
dns2.easydns.net. 165815 IN  A   198.41.222.254
dns3.easydns.ca.  107 IN  A   64.68.196.10
dns1.easydns.com. 168271 IN  AAAA 2400:cb00:2049:1::a29f:1835
dns2.easydns.net. 165815 IN  AAAA 2400:cb00:2049:1::c629:defe

;; Query time: 94 msec
;; SERVER: 192.168.1.20#53(192.168.1.20)
;; WHEN: Sun Jun 28 08:43:27 EDT 2020
;; MSG SIZE rcvd: 298

```

Đig có nhiều tùy chọn để tinh chỉnh cả đầu ra và truy vấn gửi đến máy chủ. Các tùy chọn này được bắt đầu bằng +. Một trong số đó là tùy chọn `short` sẽ chặn tất cả các đầu ra ngoại trừ kết quả:

```

$ dig +short lpi.org
65.39.134.165
$ dig +short -t SOA lpi.org
dns1.easydns.com. zone.easydns.com. 1593109612 3600 600 1209600 300

```

Dưới đây là một ví dụ về việc tắt tiện ích mở rộng EDNS của cookie:

```

$ dig +nocoookie -t MX lpi.org

; <<>> DiG 9.11.5-P4-5.1+deb10u1-Debian <<>> +nocoookie -t MX lpi.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 47774
;; flags: qr rd ra; QUERY: 1, ANSWER: 7, AUTHORITY: 3, ADDITIONAL: 5

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;lpi.org.      IN  MX

```

```
;; ANSWER SECTION:
lpi.org.      468 IN  MX  0 aspmx.l.google.com.
lpi.org.      468 IN  MX 10 aspmx4.googlemail.com.
lpi.org.      468 IN  MX 10 aspmx5.googlemail.com.
lpi.org.      468 IN  MX 10 aspmx2.googlemail.com.
lpi.org.      468 IN  MX 10 aspmx3.googlemail.com.
lpi.org.      468 IN  MX  5 alt2.aspmx.l.google.com.
lpi.org.      468 IN  MX  5 alt1.aspmx.l.google.com.

;; AUTHORITY SECTION:
lpi.org.      77130 IN  NS  dns2.easydns.net.
lpi.org.      77130 IN  NS  dns3.easydns.ca.
lpi.org.      77130 IN  NS  dns1.easydns.com.

;; ADDITIONAL SECTION:
dns1.easydns.com. 76140 IN  A   64.68.192.10
dns2.easydns.net. 73684 IN  A   198.41.222.254
dns1.easydns.com. 76140 IN  AAAA 2400:cb00:2049:1::a29f:1835
dns2.easydns.net. 73684 IN  AAAA 2400:cb00:2049:1::c629:defe

;; Query time: 2 msec
;; SERVER: 192.168.1.20#53(192.168.1.20)
;; WHEN: Mon Jun 29 10:18:58 EDT 2020
;; MSG SIZE rcvd: 389
```

Bài tập Hướng dẫn

1. Lệnh dưới đây sẽ thực hiện tác vụ gì?

```
getent group openldap
```

2. Sự khác biệt lớn nhất giữa `getent` và các công cụ `host` và `dig` là gì?

3. Tùy chọn nào cho `dig` và `host` được sử dụng để chỉ định loại bản ghi bạn muốn truy xuất?

4. Lựa chọn nào sau đây là một mục nhập `/etc/hosts` thích hợp?

```
:::1 localhost
```

```
localhost 127.0.0.1
```

5. Tùy chọn nào cho `getent` được sử dụng để chỉ định nguồn dữ liệu nào sẽ được sử dụng để thực hiện tra cứu?

Bài tập Mở rộng

1. Nếu muốn chỉnh sửa `/etc/resolv.conf` bên dưới bằng một trình soạn thảo văn bản, điều gì có thể xảy ra?

```
# Generated by NetworkManager
nameserver 192.168.1.20
```

Những thay đổi sẽ được ghi đè bởi NetworkManager.

NetworkManager sẽ cập nhật cấu hình của nó với những thay đổi của bạn.

Những thay đổi của bạn sẽ không ảnh hưởng đến hệ thống.

Trình quản lý mạng sẽ bị vô hiệu hóa.

2. Dòng sau trong `/etc/nsswitch.conf` có nghĩa là gì:

```
hosts: files [SUCCESS=continue] dns
```

3. Hãy xem `/etc/resolv.conf` sau đây: tại sao hệ thống không phân giải tên thông qua DNS?

```
search lpi.org
#nameserver fd00:ffff::1:53
#nameserver 10.0.1.53
```

4. Lệnh `dig +noall +answer +question lpi.org` sẽ thực hiện tác vụ gì?

5. Làm cách nào để có thể ghi đè các giá trị mặc định của `dig` mà không chỉ định chúng trên dòng lệnh?

Tóm tắt

Lệnh `getent` là một công cụ tuyệt vời để xem kết quả của các lệnh gọi trình phân giải. Đối với các truy vấn DNS đơn giản, `host` rất dễ sử dụng và sẽ cho ra các kết quả đơn giản. Nếu cần thông tin chi tiết hoặc cần tinh chỉnh truy vấn DNS, `dig` chính là lựa chọn tốt nhất.

Do khả năng thêm các phần bổ trợ thư viện chia sẻ và định cấu hình hành vi của trình phân giải, Linux có sự hỗ trợ tuyệt vời dành cho việc phân giải tên miền và số thuộc nhiều loại khác nhau. Chương trình `getent` có thể được sử dụng để phân giải tên miền bằng thư viện trình phân giải. `host` và `dig` có thể được sử dụng để truy vấn máy chủ DNS.

Tệp `/etc/nsswitch.conf` được sử dụng để định cấu hình hành vi của trình phân giải. Chúng ta có thể thay đổi nguồn dữ liệu và thêm một số logic điều kiện đơn giản cho các loại tên miền có nhiều nguồn.

DNS được cấu hình bằng cách chỉnh sửa `/etc/resolv.conf`. Rất nhiều bản phân phối có các công cụ quản lý tệp này nên hãy đảm bảo việc kiểm tra tài liệu hệ thống nếu các thay đổi thủ công không được duy trì.

Tệp `/etc/hosts` được sử dụng để phân giải tên máy chủ thành IP và ngược lại. Nó thường được sử dụng để xác định các tên miền (chẳng hạn như `localhost`) không có sẵn thông qua DNS.

Chúng ta có thể để lại chú thích trong các tệp cấu hình được đề cập trong bài học này. Bất kỳ một văn bản nào nằm ở bên phải ký tự `#` đều sẽ bị hệ thống bỏ qua.

Đáp án Bài tập Hướng dẫn

1. Lệnh dưới đây sẽ thực hiện tác vụ gì?

```
getent group openldap
```

Nó sẽ đọc `/etc/nsswitch.conf`, tra cứu nhóm `openldap` từ các nguồn được liệt kê và hiển thị thông tin về nhóm nếu tìm thấy.

2. Sự khác biệt lớn nhất giữa `getent` và các công cụ `host` và `dig` là gì?

`getent` sẽ tra cứu tên miền bằng thư viện trình phân giải, các công cụ còn lại sẽ chỉ truy vấn DNS. `getent` có thể được sử dụng để khắc phục sự cố `/etc/nsswitch.conf` và cấu hình thư viện phân giải tên miền mà hệ thống được định cấu hình để sử dụng. `host` và `dig` được sử dụng để tra cứu bản ghi DNS.

3. Tùy chọn nào cho `dig` và `host` được sử dụng để chỉ định loại bản ghi bạn muốn truy xuất?

Cả hai chương trình đều sử dụng `-t` để chỉ định loại bản ghi cần tra cứu.

4. Lựa chọn nào sau đây là một mục nhập `/etc/hosts` thích hợp?

```
:::1 localhost
```

```
X
```

```
localhost 127.0.0.1
```

`:::1 localhost` là dòng đúng. Cột bên trái luôn là địa chỉ IPv4 hoặc IPv6.

5. Tùy chọn nào cho `getent` được sử dụng để chỉ định nguồn dữ liệu nào sẽ được sử dụng để thực hiện tra cứu?

Tùy chọn `-s` được sử dụng để chỉ định nguồn dữ liệu. Ví dụ:

```
$ getent -s files hosts learning.lpi.org
192.168.10.25 learning.lpi.org
$ getent -s dns hosts learning.lpi.org
208.94.166.198 learning.lpi.org
```

Đáp án Bài tập Mở rộng

1. Nếu muốn chỉnh sửa `/etc/resolv.conf` bên dưới bằng một trình soạn thảo văn bản, điều gì có thể xảy ra?

```
# Generated by NetworkManager
nameserver 192.168.1.20
```

Những thay đổi sẽ được ghi đè bởi NetworkManager.	X
NetworkManager sẽ cập nhật cấu hình của nó với những thay đổi của bạn.	
Những thay đổi của bạn sẽ không ảnh hưởng đến hệ thống.	
Trình quản lý mạng sẽ bị vô hiệu hóa.	

2. Dòng sau trong `/etc/nsswitch.conf` có nghĩa là gì:

```
hosts: files [SUCCESS=continue] dns
```

Việc tra cứu tên máy chủ sẽ kiểm tra các tệp `/etc/hosts` của bạn trước rồi đến DNS. Nếu tìm thấy một mục nhập trong tệp và DNS, mục nhập đó trong DNS sẽ được sử dụng.

3. Hãy xem `/etc/resolv.conf` sau đây: tại sao hệ thống không phân giải tên miền thông qua DNS?

```
search lpi.org
#nameserver fd00:ffff::1:53
#nameserver 10.0.1.53
```

Cả hai máy chủ DNS đều được chú thích và không có máy chủ DNS nào đang chạy trên máy chủ cục bộ.

4. Lệnh `dig +noall +answer +question lpi.org` sẽ thực hiện tác vụ gì?

Nó tra cứu bản ghi A cho `lpi.org` và chỉ hiển thị truy vấn và phản hồi.

5. Làm cách nào để có thể ghi đè các giá trị mặc định của `dig` mà không chỉ định chúng trên dòng

lệnh?

Hãy tạo một tệp `.digrc` trong thư mục chính của mình.



Chủ đề 110: Bảo mật



110.1 Thực hiện nhiệm vụ Quản trị Bảo mật

Tham khảo các mục tiêu LPI

[LPIC-1 version 5.0, Exam 102, Objective 110.1](#)

Khối lượng

3

Các lĩnh vực kiến thức chính

- Kiểm duyệt một hệ thống để tìm các tệp có tập bit suid/sgid.
- Đặt hoặc thay đổi mật khẩu người dùng và thông tin lão hoá mật khẩu.
- Có khả năng sử dụng nmap và netstat để phát hiện các cổng đang mở trên hệ thống.
- Thiết lập giới hạn về thông tin đăng nhập, quy trình và mức sử dụng bộ nhớ của người dùng.
- Xác định người dùng nào đã đăng nhập vào hệ thống hoặc hiện đang đăng nhập.
- Cấu hình và cách sử dụng sudo cơ bản.

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- `find`
- `passwd`
- `fuser`
- `lsof`
- `nmap`
- `chage`
- `netstat`
- `sudo`

- `/etc/sudoers`
- `su`
- `usermod`
- `ulimit`
- `who, w, last`



110.1 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	110 Bảo mật
Mục tiêu:	110.1 Thực hiện nhiệm vụ Quản trị Bảo mật
Bài:	1 trên 1

Giới thiệu

Bảo mật là một yếu tố bắt buộc trong quản trị hệ thống. Là một quản trị viên hệ thống Linux hiệu quả, người dùng phải để mắt đến một số khía cạnh như quyền đặc biệt trên tệp, tuổi thọ mật khẩu người dùng, cổng và ổ nối mở, hạn chế sử dụng tài nguyên hệ thống, xử lý những người dùng đang đăng nhập và nâng cao đặc quyền thông qua `su` và `sudo`. Trong bài học này, chúng ta sẽ cùng ôn lại tất cả những chủ đề này.

Kiểm tra tệp bằng bộ SUID và SGID

Ngoài bộ quyền truyền thống gồm *đọc*, *ghi* và *thực thi*, các tệp trong hệ thống Linux cũng có thể có một số bộ quyền đặc biệt như các bit *SUID* hoặc *SGID*.

Bit SUID sẽ cho phép tệp được thực thi với đặc quyền của chủ sở hữu tệp. Nó được biểu thị dưới dạng số bằng `4000` và được biểu thị dưới dạng ký hiệu bằng `s` hoặc `S` trên bit quyền *thực thi* của chủ sở hữu. Một ví dụ kinh điển về tệp thực thi có bộ quyền SUID là `passwd`:

```
carol@debian:~$ ls -l /usr/bin/passwd
```



```
-rwsr-xr-x 1 root root 63736 jul 27 2018 /usr/bin/passwd
```

Chữ `s` viết thường trong `rws` biểu thị sự hiện diện của SUID trên tệp — cùng với quyền *thực thi*. Nếu là một chữ `S` viết hoa (`rwS`) thì nó sẽ có nghĩa là quyền *thực thi* cơ bản không được đặt.

NOTE

Bạn sẽ tìm hiểu về `passwd` trong phần tiếp theo. Tiện ích này chủ yếu được `root` sử dụng để đặt/thay đổi mật khẩu của người dùng (ví dụ: `passwd carol`). Tuy nhiên, người dùng thông thường cũng có thể sử dụng nó để thay đổi mật khẩu của riêng họ, vì thế mà nó có đi kèm với bộ SUID.

Mật khác, bit SGID có thể được đặt trên cả tệp và thư mục. Với các tệp, hoạt động của nó cũng tương đương với SUID nhưng các đặc quyền sẽ thuộc về chủ sở hữu nhóm. Tuy nhiên, khi được đặt trên một thư mục, nó sẽ cho phép tất cả các tệp được tạo trong đó kế thừa quyền sở hữu của nhóm thư mục. Giống như SUID, SGID cũng được biểu thị một cách tượng trưng bằng `s` hoặc `S` trên bit quyền *thực thi* của nhóm. Dưới dạng số, nó được biểu thị bằng `2000`. Chúng ta có thể đặt SGID trên một thư mục bằng cách sử dụng `chmod` và thêm `2` (SGID) vào các quyền truyền thống (trong trường hợp của chúng ta là `755`):

```
carol@debian:~$ ls -ld shared_directory
drwxr-xr-x 2 carol carol 4096 may 30 23:55 shared_directory
carol@debian:~$ sudo chmod 2755 shared_directory/
carol@debian:~$ ls -ld shared_directory
drwxr-sr-x 2 carol carol 4096 may 30 23:55 shared_directory
```

Để tìm các tệp có một hoặc cả hai bộ SUID và SGID, chúng ta có thể sử dụng lệnh `find` và sử dụng tùy chọn `-perm`. Ta có thể sử dụng cả giá trị số và ký hiệu. Các giá trị cũng có thể được truyền riêng hoặc đứng sau một dấu gạch ngang (`-`) hoặc gạch chéo lên (`/`). Ý nghĩa của chúng sẽ như sau:

`-perm numeric-value` hoặc `-perm symbolic-value`

tìm các tệp có quyền đặc biệt *độc nhất*

`-perm -numeric-value` hoặc `-perm -symbolic-value`

tìm các tệp có quyền đặc biệt và các quyền khác

`-perm /numeric-value` hoặc `-perm /symbolic-value`

tìm các tệp có quyền đặc biệt (và các quyền khác)

Ví dụ: để tìm các tệp *chỉ* có SUID được đặt trong thư mục làm việc hiện tại, chúng ta sẽ sử dụng lệnh sau:

```
carol@debian:~$ find . -perm 4000
carol@debian:~$ touch file
carol@debian:~$ chmod 4000 file
carol@debian:~$ find . -perm 4000
./file
```

Vì không có bất kỳ tệp nào chỉ có SUID, hãy lưu ý việc chúng ta đã tạo một tệp để hiển thị một phần đầu ra. Chúng ta có thể chạy lệnh tương tự bằng ký hiệu tượng trưng:

```
carol@debian:~$ find . -perm u+s
./file
```

Để tìm các tệp khớp với SUID (bất kể bất kỳ một quyền nào khác) trong thư mục `/usr/bin/`, chúng ta có thể sử dụng một trong các lệnh sau:

```
carol@debian:~$ sudo find /usr/bin -perm -4000
/usr/bin/umount
/usr/bin/newgrp
/usr/bin/gpasswd
/usr/bin/chfn
/usr/bin/mount
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/sudo
/usr/bin/su
carol@debian:~$ sudo find /usr/bin -perm -u+s
/usr/bin/umount
/usr/bin/newgrp
/usr/bin/gpasswd
/usr/bin/chfn
/usr/bin/mount
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/sudo
/usr/bin/su
```

Nếu đang tìm các tệp trong cùng một thư mục với tập bit SGID, chúng ta có thể thực thi `find /usr/bin/ -perm -2000` hoặc `find /usr/bin/ -perm -g+s`.

Cuối cùng, để tìm các tệp có một trong hai bộ quyền đặc biệt, hãy thêm 4 và 2 và sử dụng `/:`

```
carol@debian:~$ sudo find /usr/bin -perm /6000
/usr/bin/dotlock.mailutils
/usr/bin/umount
/usr/bin/newgrp
/usr/bin/wall
/usr/bin/ssh-agent
/usr/bin/chage
/usr/bin/dotlockfile
/usr/bin/gpasswd
/usr/bin/chfn
/usr/bin/mount
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/expiry
/usr/bin/sudo
/usr/bin/bsd-write
/usr/bin/crontab
/usr/bin/su
```

Quản lý Mật khẩu và Tuổi thọ Mật khẩu

Như đã được chú thích ở trên, chúng ta có thể sử dụng tiện ích `passwd` để thay đổi mật khẩu của riêng mình với tư cách là người dùng thông thường. Ngoài ra, chúng ta cũng có thể truyền khoá chuyển `-S` hoặc `--status` để nhận thông tin trạng thái về tài khoản của mình:

```
carol@debian:~$ passwd -S
carol P 12/07/2019 0 99999 7 -1
```

Dưới đây là một bảng phân tích về bẫy trường chúng ta sẽ nhận được trong đầu ra:

carol

Tên đăng nhập của người dùng.

P

Chỉ ra rằng người dùng có mật khẩu hợp lệ (P); các giá trị có thể có khác là L cho mật khẩu bị khóa (locked) và NP cho trạng thái không có mật khẩu (no password).

12/07/2019

Ngày thay đổi mật khẩu cuối cùng.

0

Tuổi tối thiểu tính theo ngày (số ngày tối thiểu giữa các lần thay đổi mật khẩu). Giá trị 0 có nghĩa là mật khẩu có thể được thay đổi bất cứ lúc nào.

99999

Tuổi tối đa tính theo ngày (số ngày tối đa mà mật khẩu có hiệu lực). Giá trị 99999 sẽ vô hiệu hóa việc hết hạn mật khẩu.

7

Khoảng thời gian cảnh báo tính bằng ngày (số ngày trước khi mật khẩu hết hạn mà người dùng sẽ được cảnh báo).

-1

Khoảng thời gian mật khẩu không hoạt động tính bằng ngày (số ngày không hoạt động sau khi mật khẩu hết hạn và trước khi tài khoản bị khóa). Giá trị -1 sẽ loại bỏ trạng thái không hoạt động của tài khoản.

Ngoài việc báo cáo về trạng thái tài khoản, chúng ta còn có thể sử dụng lệnh `passwd` dưới tên siêu người dùng để thực hiện một số thao tác bảo trì tài khoản cơ bản. Chúng ta có thể khóa (lock) và mở khóa tài khoản (unlock), buộc người dùng thay đổi (edit) mật khẩu trong lần đăng nhập tiếp theo và xóa (delete) mật khẩu của người dùng bằng các tùy chọn `-l`, `-u`, `-e` và `-d` tương ứng.

Đúng lúc cần kiểm tra các tùy chọn này, chúng ta có thể đồng thời thuận tiện tìm hiểu về lệnh `su`. Thông qua `su`, chúng ta có thể thay đổi người dùng trong phiên đăng nhập. Ví dụ: hãy cùng sử dụng `passwd` dưới tên siêu người dùng để khóa mật khẩu của `carol`. Sau đó, chúng ta sẽ chuyển sang người dùng `carol` và kiểm tra trạng thái tài khoản của mình để xác minh rằng mật khẩu — trên thực tế — đã bị khóa (L) và không thể thay đổi. Cuối cùng là quay lại siêu người dùng và mở khóa mật khẩu của người dùng `carol`.

```
root@debian:~# passwd -l carol
passwd: password expiry information changed.
root@debian:~# su - carol
carol@debian:~$ passwd -S
carol L 05/31/2020 0 99999 7 -1
carol@debian:~$ passwd
Changing password for carol.
Current password:
passwd: Authentication token manipulation error
passwd: password unchanged
carol@debian:~$ exit
logout
```

```
root@debian:~# passwd -u carol
passwd: password expiry information changed.
```

Ngoài ra, chúng ta cũng có thể khóa và mở khóa mật khẩu của người dùng bằng lệnh `usermod`:

Khoá mật khẩu của `carol`

```
usermod -L carol hoặc usermod --lock carol.
```

Mở khoá mật khẩu của `carol`

```
usermod -U carol hoặc usermod --unlock carol.
```

NOTE

Với các khoá chuyển `-f` hoặc `--inactive`, `usermod` cũng có thể được sử dụng để đặt số ngày trước khi tài khoản có mật khẩu hết hạn bị vô hiệu hóa (ví dụ: `usermod -f 3 carol`).

Ngoài `passwd` và `usermod`, lệnh trực tiếp nhất để xử lý vấn đề lão hóa mật khẩu và tài khoản là `chage` (“change age”). Với quyền gốc, chúng ta có thể truyền cho `chage` khoá chuyển `-l` (hoặc `--list`), theo sau là tên người dùng để in mật khẩu hiện tại của người dùng đó và thông tin hết hạn tài khoản trên màn hình; với tư cách là người dùng thông thường, chúng ta có thể xem thông tin của chính mình:

```
carol@debian:~$ chage -l carol
Last password change           : Aug 06, 2019
Password expires               : never
Password inactive              : never
Account expires                : never
Minimum number of days between password change : 0
Maximum number of days between password change : 99999
Number of days of warning before password expires : 7
```

Nếu phải chạy mà không có tùy chọn và chỉ có tên người dùng theo sau, `chage` sẽ hoạt động một cách tương tác:

```
root@debian:~# chage carol
Changing the aging information for carol
Enter the new value, or press ENTER for the default

Minimum Password Age [0]:
Maximum Password Age [99999]:
Last Password Change (YYYY-MM-DD) [2020-06-01]:
Password Expiration Warning [7]:
```

```
Password Inactive [-1]:
Account Expiration Date (YYYY-MM-DD) [-1]:
```

Các tùy chọn để sửa đổi các cài đặt chage khác nhau như sau:

-m days username hoặc **--mindays days username**

Chỉ định số ngày tối thiểu giữa các lần thay đổi mật khẩu (ví dụ: `chage -m 5 carol`). Giá trị 0 sẽ cho phép người dùng thay đổi mật khẩu của mình bất cứ lúc nào.

-M days username hoặc **--maxdays days username**

Chỉ định số ngày tối đa mà mật khẩu sẽ có hiệu lực (ví dụ: `chage -M 30 carol`). Để vô hiệu hóa việc hết hạn mật khẩu, hãy đặt giá trị cho tùy chọn này là 99999.

-d days username hoặc **--lastday days username**

Chỉ định số ngày kể từ lần thay đổi mật khẩu lần cuối (ví dụ: `chage -d 10 carol`). Giá trị 0 sẽ buộc người dùng thay đổi mật khẩu vào lần đăng nhập tiếp theo.

-W days username hoặc **--warndays days username**

Chỉ định số ngày người dùng sẽ được nhắc về việc mật khẩu của họ đã hết hạn.

-I days username hoặc **--inactive days username**

Chỉ định số ngày không hoạt động sau khi hết hạn mật khẩu (ví dụ: `chage -I 10 carol`)—giống như `usermod -f` hoặc `usermod --inactive`. Sau số ngày đó, tài khoản sẽ bị khóa. Tuy nhiên, với giá trị 0, tài khoản sẽ không bị khóa.

-E date username hoặc **--expiredate date username**

Chỉ định ngày (hoặc số ngày kể từ *Kỷ nguyên*—ngày 1 tháng 1 năm 1970) mà tài khoản sẽ bị khóa. Nó thường được biểu thị ở định dạng YYYY-MM-DD (ví dụ: `chage -E 2050-12-13 carol`).

NOTE

Bạn có thể tìm hiểu thêm về `passwd`, `usermod` và `chage`—và các tùy chọn của chúng—bằng cách tham khảo các trang hướng dẫn tương ứng.

Khám phá các Cổng mở

Khi nói đến việc theo dõi các cổng đang mở, có bốn tiện ích mạnh mẽ đều có mặt trên hầu hết các hệ thống Linux: `lsof`, `fuser`, `netstat` và `nmap`. Chúng ta sẽ cùng tìm hiểu về chúng trong phần này.

`lsof` là viết tắt của “list open files” (liệt kê các tệp đang mở). Đây không phải là vấn đề nhỏ vì—đối với Linux—mọi thứ trong hệ thống đều là một tệp. Trên thực tế, nếu gõ `lsof` vào cửa sổ

dòng lệnh, chúng ta sẽ nhận được một danh sách lớn các tệp thông thường, tệp thiết bị, ổ nối, v.v. Tuy nhiên, trong bài học này, chúng ta sẽ chủ yếu tập trung vào các cổng. Để in danh sách tất cả các tệp mạng “Internet”, hãy chạy `lsof` với tùy chọn `-i`:

```
root@debian:~# lsof -i
COMMAND PID    USER  FD  TYPE DEVICE SIZE/OFF NODE NAME
dhclient 357    root   7u  IPv4 13493      0t0  UDP *:bootpc
sshd     389    root   3u  IPv4 13689      0t0  TCP *:ssh (LISTEN)
sshd     389    root   4u  IPv6 13700      0t0  TCP *:ssh (LISTEN)
apache2  399    root   3u  IPv6 13826      0t0  TCP *:http (LISTEN)
apache2  401  www-data 3u  IPv6 13826      0t0  TCP *:http (LISTEN)
apache2  402  www-data 3u  IPv6 13826      0t0  TCP *:http (LISTEN)
sshd     557    root   3u  IPv4 14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
sshd     569    carol   3u  IPv4 14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
```

Ngoài dịch vụ `bootpc` — được DHCP sử dụng — đầu ra sẽ còn hiển thị hai dịch vụ đang lắng nghe kết nối — `ssh` và máy chủ web Apache (`http`) — cũng như hai kết nối SSH đã thiết lập. Chúng ta có thể chỉ định một máy chủ cụ thể bằng ký hiệu `@ip-address` để kiểm tra các kết nối của nó:

```
root@debian:~# lsof -i@192.168.1.7
COMMAND PID  USER  FD  TYPE DEVICE SIZE/OFF NODE NAME
sshd     557  root   3u  IPv4 14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
sshd     569  carol   3u  IPv4 14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
```

NOTE Để chỉ in các tệp mạng IPv4 và IPv6, hãy sử dụng tùy chọn `-i4` và `-i6` tương ứng.

Tương tự, chúng ta cũng có thể lọc theo cổng bằng cách truyền cho tùy chọn `-i` (hoặc `-i@ip-address`) đối số `:port`:

```
root@debian:~# lsof -i :22
COMMAND PID  USER  FD  TYPE DEVICE SIZE/OFF NODE NAME
sshd     389  root   3u  IPv4 13689      0t0  TCP *:ssh (LISTEN)
sshd     389  root   4u  IPv6 13700      0t0  TCP *:ssh (LISTEN)
sshd     557  root   3u  IPv4 14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
sshd     569  carol   3u  IPv4 14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
```

Các cổng sẽ được phân tách bằng dấu phẩy (và phạm vi sẽ được chỉ định bằng dấu gạch ngang):

```
root@debian:~# lsof -i@192.168.1.7:22,80
COMMAND PID  USER  FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd     705  root   3u   IPv4  13960      0t0  TCP 192.168.1.7:ssh->192.168.1.4:44766
(ESTABLISHED)
sshd     718  carol  3u   IPv4  13960      0t0  TCP 192.168.1.7:ssh->192.168.1.4:44766
(ESTABLISHED)
```

NOTE

`lsof` có một số lượng tùy chọn có sẵn khá đáng kể. Để tìm hiểu thêm, hãy tham khảo trang hướng dẫn của nó.

Tiếp theo trong danh sách các lệnh về mạng là `fuser`. Mục đích chính của nó là tìm “người dùng của tệp” — bao gồm việc biết các tiến trình nào đang truy cập vào tệp nào; nó cũng cung cấp cho người dùng một số thông tin khác như loại quyền truy cập. Ví dụ: để kiểm tra thư mục làm việc hiện tại, chúng ta chỉ cần chạy `fuser` là đủ. Tuy nhiên, để biết thêm thông tin, chúng ta có thể sử dụng tùy chọn đầu ra chi tiết (`-v` hoặc `--verbose`):

```
root@debian:~# fuser .
/root:                    580c
root@debian:~# fuser -v .
                USER      PID ACCESS COMMAND
/root:          root      580  .c.. bash
```

Hãy cùng chia nhỏ đầu ra:

Tệp

Tệp mà chúng ta đang muốn nhận thông tin (`/root`).

Cột USER

Chủ sở hữu của tệp (`root`).

Cột PID

Mã định danh tiến trình (`580`).

Cột ACCESS

Loại truy cập (`. . c . .`). Một trong số đó:

c

Thư mục hiện tại.

e

Có thể thực thi được.

f

Mở tệp (bỏ qua ở chế độ hiển thị mặc định).

F

Mở tệp để ghi (bỏ qua ở chế độ hiển thị mặc định).

r

Thư mục gốc.

m

Tệp mmap hoặc thư viện chia sẻ

.

Ký tự giữ chỗ (bỏ qua trong chế độ hiển thị mặc định).

Cột COMMAND

Lệnh liên kết với tệp (bash).

Với tùy chọn `-n` (hoặc `--namespace`), chúng ta có thể tìm thông tin về các cổng/ổ nối mạng. Ta sẽ phải cung cấp giao thức mạng và số cổng. Vì vậy, để lấy thông tin về máy chủ web Apache, chúng ta sẽ chạy lệnh sau:

```
root@debian:~# fuser -vn tcp 80
      USER      PID ACCESS COMMAND
80/tcp:      root         402 F.... apache2
           www-data    404 F.... apache2
           www-data    405 F.... apache2
```

NOTE

`fuser` cũng có thể được sử dụng để hủy các tiến trình đang truy cập tệp bằng các khóa chuyển `-k` hoặc `--kill` (ví dụ: `fuser -k 80/tcp`). Hãy tham khảo trang hướng dẫn để biết thêm thông tin chi tiết.

Bây giờ chúng ta sẽ chuyển sang `netstat`. `netstat` là một công cụ mạng rất linh hoạt, chủ yếu được sử dụng để in “số liệu thống kê về mạng”.

Nếu được chạy mà không có tùy chọn, `netstat` sẽ hiển thị cả kết nối Internet đang hoạt động và ổ nối Unix. Do kích thước của danh sách, chúng ta có thể sẽ muốn dẫn ống đầu ra của nó thông qua `less`:

```

carol@debian:~$ netstat |less
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 192.168.1.7:ssh        192.168.1.4:55444     ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags           Type           State           I-Node  Path
unix   2      [ ]           DGRAM                    10509  /run/systemd/journal/syslog
unix   3      [ ]           DGRAM                    10123  /run/systemd/notify
(...)

```

Để chỉ liệt kê các cổng và ổ nối “đang nghe”, chúng ta sẽ sử dụng tùy chọn `-l` hoặc `--listening`. Các tùy chọn `-t/--tcp` và `-u/--udp` có thể được thêm vào để tương ứng lọc theo giao thức TCP và UDP (chúng cũng có thể được kết hợp trong cùng một lệnh). Tương tự, `-e/--extend` sẽ hiển thị các thông tin bổ sung:

```

carol@debian:~$ netstat -lu
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
udp      0      0 0.0.0.0:bootpc        0.0.0.0:*
carol@debian:~$ netstat -lt
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 0.0.0.0:ssh           0.0.0.0:*              LISTEN
tcp      0      0 localhost:smtp        0.0.0.0:*              LISTEN
tcp6     0      0 [::]:http            [::]:*                 LISTEN
tcp6     0      0 [::]:ssh             [::]:*                 LISTEN
tcp6     0      0 localhost:smtp       [::]:*                 LISTEN
carol@debian:~$ netstat -lute
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State   User
Inode
tcp      0      0 0.0.0.0:ssh           0.0.0.0:*              LISTEN root
13729
tcp      0      0 localhost:smtp        0.0.0.0:*              LISTEN root
14372
tcp6     0      0 [::]:http            [::]:*                 LISTEN root
14159
tcp6     0      0 [::]:ssh             [::]:*                 LISTEN root
13740
tcp6     0      0 localhost:smtp       [::]:*                 LISTEN root
14374
udp      0      0 0.0.0.0:bootpc        0.0.0.0:*

```

13604

Nếu bỏ qua tùy chọn `-l` thì *chỉ riêng* các kết nối đã được thiết lập mới được hiển thị:

```
carol@debian:~$ netstat -ute
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       User
Inode
tcp        0      0 192.168.1.7:ssh        192.168.1.4:39144      ESTABLISHED root
15103
```

Nếu chỉ quan tâm đến thông tin về các con số liên quan đến cổng và máy chủ, chúng ta có thể sử dụng tùy chọn `-n` hoặc `--numeric` để chỉ in số cổng và địa chỉ IP. Hãy lưu ý việc `ssh` biến thành `22` khi thêm `-n` vào lệnh trên:

```
carol@debian:~$ netstat -uten
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       User
Inode
tcp        0      0 192.168.1.7:22        192.168.1.4:39144      ESTABLISHED 0
15103
```

Như có thể thấy được, chúng ta có thể tạo các lệnh `netstat` rất hữu ích và hiệu quả bằng cách kết hợp một số tùy chọn của nó. Hãy xem qua các trang hướng dẫn để tìm hiểu thêm và tìm ra sự kết hợp phù hợp nhất đối với từng nhu cầu.

Cuối cùng, chúng ta sẽ tìm hiểu về `nmap` — hoặc “trình ánh xạ mạng”. Trình quét cổng này là một tiện ích rất mạnh mẽ khác được thực thi bằng cách chỉ định địa chỉ IP hoặc tên máy chủ:

```
root@debian:~# nmap localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2020-06-04 19:29 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000040s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 1.58 seconds
```

Ngoài một máy chủ duy nhất, `nmap` cũng cho phép chúng ta quét:

nhều máy chủ

bằng cách phân tách chúng bằng dấu cách (ví dụ: `nmap localhost 192.168.1.7`).

các phạm vi máy chủ

bằng cách sử dụng dấu gạch ngang (ví dụ: `nmap 192.168.1.3-20`).

các mạng con

bằng cách sử dụng ký hiệu đại diện hoặc ký hiệu CIDR (ví dụ: `nmap 192.168.1.*` hoặc `nmap 192.168.1.0/24`). Chúng ta cũng có thể loại trừ các máy chủ cụ thể (ví dụ: `nmap 192.168.1.0/24 --exclude 192.168.1.7`).

Để quét một cổng cụ thể, hãy sử dụng khóa chuyển `-p`, theo sau là số cổng hoặc tên dịch vụ (`nmap -p 22` và `nmap -p ssh` sẽ cho ta cùng một đầu ra):

```
root@debian:~# nmap -p 22 localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2020-06-04 19:54 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000024s latency).
Other addresses for localhost (not scanned): ::1

PORT      STATE SERVICE
22/tcp    open  ssh

Nmap done: 1 IP address (1 host up) scanned in 0.22 seconds
```

Chúng ta cũng có thể quét nhiều cổng hoặc phạm vi cổng bằng cách sử dụng tương ứng dấu phẩy và dấu gạch ngang:

```
root@debian:~# nmap -p ssh,80 localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2020-06-04 19:58 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000051s latency).
Other addresses for localhost (not scanned): ::1

PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 0.22 seconds
```

```

root@debian:~# nmap -p 22-80 localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2020-06-04 19:58 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000011s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 57 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 1.47 seconds

```

Hai tùy chọn `nmap` rất quan trọng và tiện dụng khác là:

-F

Chạy một lượt quét nhanh trên 100 cổng phổ biến nhất.

-v

Nhận đầu ra chi tiết (`-vv` sẽ in đầu ra chi tiết hơn nữa).

NOTE

`nmap` có thể chạy các lệnh khá phức tạp bằng cách tận dụng các kiểu quét. Tuy nhiên, chủ đề đó nằm ngoài phạm vi của bài học này.

Giới hạn về Thông tin đăng nhập của Người dùng, Tiến trình và mức sử dụng Bộ nhớ

Tài nguyên trên hệ thống Linux không phải là vô hạn. Vì vậy—với tư cách là quản trị viên hệ thống—người dùng nên đảm bảo được sự cân bằng giữa *giới hạn người dùng* về tài nguyên và tình trạng hoạt động bình thường của hệ điều hành. `ulimit` có thể hỗ trợ chúng ta trên phương diện này.

`ulimit` xử lý các giới hạn *mềm* (soft) và *cứng* (hard)—được chỉ định tương ứng bởi các tùy chọn `-S` và `-H`. Nếu được chạy mà không có tùy chọn hoặc đối số, `ulimit` sẽ hiển thị các khối tệp mềm của người dùng hiện tại:

```

carol@debian:~$ ulimit
unlimited

```

Với tùy chọn `-a`, `ulimit` sẽ hiển thị tất cả các giới hạn mềm hiện tại (giống như `-Sa`); để hiển thị tất cả các giới hạn cứng hiện tại, hãy sử dụng `-Ha`:

```

carol@debian:~$ ulimit -a
core file size          (blocks, -c) 0
data seg size          (kbytes, -d) unlimited
scheduling priority    (-e) 0
(...)
carol@debian:~$ ulimit -Ha
core file size          (blocks, -c) unlimited
data seg size          (kbytes, -d) unlimited
scheduling priority    (-e) 0
(...)

```

Các tài nguyên vổ có sẵn sẽ được chỉ định bởi các tùy chọn như:

-b

kích thước bộ đệm ổ nổi tối đa

-f

kích thước tối đa của tệp được ghi bởi vổ và các vổ con

-l

kích thước tối đa có thể bị khóa trong bộ nhớ

-m

kích thước cài đặt thường trú tối đa (RSS) — phần bộ nhớ hiện tại được nắm giữ bởi một tiến trình trong bộ nhớ chính (RAM)

-v

dung lượng bộ nhớ ảo tối đa

-u

số lượng tiến trình tối đa có sẵn cho một người dùng

Do đó, để hiển thị giới hạn, chúng ta sẽ sử dụng `ulimit`, theo sau là `-S` (mềm) hoặc `-H` (cứng) và tùy chọn tài nguyên; nếu cả `-S` hoặc `-H` đều không được cung cấp, các giới hạn mềm sẽ được hiển thị:

```

carol@debian:~$ ulimit -u
10000
carol@debian:~$ ulimit -Su
10000
carol@debian:~$ ulimit -Hu

```

15672

Tương tự, để đặt các giới hạn mới cho một tài nguyên cụ thể, chúng ta sẽ chỉ định `-S` hoặc `-H`, theo sau là tùy chọn tài nguyên tương ứng và giá trị mới. Giá trị này có thể là một chữ số hoặc các từ đặc biệt là `soft` (giới hạn mềm hiện tại), `hard` (giới hạn cứng hiện tại) hoặc `unlimited` (không giới hạn). Nếu cả `-S` hoặc `-H` đều không được chỉ định, cả hai giới hạn sẽ được đặt. Ví dụ: trước tiên chúng ta sẽ đọc giá trị kích thước tối đa hiện tại cho các tệp được ghi bởi vỏ và các vỏ con của nó:

```
root@debian:~# ulimit -Sf
unlimited
root@debian:~# ulimit -Hf
unlimited
```

Bây giờ, chúng ta sẽ thay đổi giá trị từ `unlimited` thành các khối `500` mà không chỉ định `-S` hoặc `-H`. Hãy lưu ý việc cả giới hạn mềm và cứng đều thay đổi:

```
root@debian:~# ulimit -f 500
root@debian:~# ulimit -Sf
500
root@debian:~# ulimit -Hf
500
```

Cuối cùng, chúng ta sẽ chỉ giảm giới hạn mềm xuống các khối `200`:

```
root@debian:~# ulimit -Sf 200
root@debian:~# ulimit -Sf
200
root@debian:~# ulimit -Hf
500
```

Giới hạn cứng sẽ chỉ có thể được tăng lên bởi siêu người dùng. Mặt khác, người dùng thông thường có thể giảm giới hạn cứng và tăng giới hạn mềm lên thành giá trị giới hạn cứng. Để duy trì các giá trị giới hạn mới trong các lần khởi động lại, chúng ta phải ghi chúng vào tệp `/etc/security/limits.conf`. Đây cũng là tệp được quản trị viên sử dụng để áp dụng các hạn chế đối với những người dùng cụ thể.

NOTE

Hãy chú ý rằng không có trang hướng dẫn dành riêng cho `ulimit` như các lệnh khác. Nó là một bash tích hợp sẵn nên bạn sẽ phải tham khảo trang hướng dẫn của

bash để tìm hiểu về nó.

Xử lý những Người dùng đã đăng nhập

Một công việc khác của người dùng với tư cách quản trị viên hệ thống chính là theo dõi những người dùng đã đăng nhập. Có ba tiện ích có thể giúp chúng ta thực hiện công việc này là `last`, `who` và `w`.

`last` sẽ in danh sách những người dùng đăng nhập gần đây nhất với thông tin mới nhất ở trên cùng:

```
root@debian:~# last
carol pts/0 192.168.1.4 Sat Jun 6 14:25 still logged in
reboot system boot 4.19.0-9-amd64 Sat Jun 6 14:24 still running
mimi pts/0 192.168.1.4 Sat Jun 6 12:07 - 14:24 (02:16)
reboot system boot 4.19.0-9-amd64 Sat Jun 6 12:07 - 14:24 (02:17)
(...)
wtmp begins Sun May 31 14:14:58 2020
```

Nhìn vào danh sách rút gọn, chúng ta đã nhận được thông tin về hai người dùng cuối cùng trên hệ thống. Hai dòng đầu tiên cho chúng ta biết về người dùng `carol`, hai dòng tiếp theo là về người dùng `mimi`. Thông tin sẽ như sau:

1. Người dùng `carol` tại cửa sổ dòng lệnh `pts/0` từ máy chủ `192.168.1.4` đã bắt đầu phiên của mình vào `Sat Jun 6` (Thứ Bảy ngày 6 tháng 6) lúc `14:25` và vẫn đang `logged in` (đăng nhập). Hệ thống — sử dụng hạt nhân `4.19.0-9-amd64` — đã được khởi động (`reboot system boot`) vào `Sat Jun 6` lúc `14:24` và `still running` (vẫn đang chạy).
2. Người dùng `mimi` tại cửa sổ dòng lệnh `pts/0` từ máy chủ `192.168.1.4` đã bắt đầu phiên của mình vào `Sat Jun 6` lúc `12:07` và đăng xuất lúc `14:24` (phiên kéo dài tổng cộng `(02:16)` giờ). Hệ thống — sử dụng hạt nhân `4.19.0-9-amd64` — đã được khởi động (`reboot system boot`) vào `Sat Jun 6` lúc `12:07` và bị tắt lúc `14:24` (nó đã hoạt động được `(02:17)` giờ).

NOTE

Dòng `wtmp begins Sun May 31 14:14:58 2020` đề cập đến `/var/log/wtmp` - tệp nhật ký đặc biệt mà `last` đã lấy thông tin.

Chúng ta có thể truyền cho `last` một tên người dùng để chỉ hiển thị các mục nhập cho người dùng đó:

```
root@debian:~# last carol
carol pts/0 192.168.1.4 Sat Jun 6 14:25 still logged in
carol pts/0 192.168.1.4 Sat Jun 6 12:07 - 14:24 (02:16)
```



```
carol pts/0 192.168.1.4 Fri Jun 5 00:48 - 01:28 (00:39)
(...)
```

Về cột thứ hai (cửa sổ dòng lệnh), `pts` là viết tắt của *Pseudo Terminal Slave* (Trình mô phỏng Cửa sổ Dòng lệnh đồ họa) — trái ngược với một cửa sổ dòng lệnh *TeleTYpewriter* (hoặc `tty`) hoàn chỉnh. `0` tức là cửa sổ dòng lệnh đầu tiên (số đếm bắt đầu từ 0).

NOTE Để kiểm tra những lần đăng nhập lỗi, hãy chạy `lastb` thay vì `last`.

Tiện ích `who` và `w` khá giống nhau và đều tập trung vào người dùng hiện đã đăng nhập. `who` sẽ hiển thị ai đã đăng nhập, trong khi `w` ngoài ra còn hiển thị thông tin về những gì họ đang làm.

Khi được thực thi mà không có tùy chọn nào, `who` sẽ hiển thị bốn cột tương ứng với người dùng, cửa sổ dòng lệnh, ngày giờ và tên máy chủ đã đăng nhập:

```
root@debian:~# who
carol pts/0 2020-06-06 17:16 (192.168.1.4)
mimi pts/1 2020-06-06 17:28 (192.168.1.4)
```

`who` chấp nhận một loạt các tùy chọn, trong số đó có các tùy chọn nổi bật sau:

-b,--boot

Hiển thị thời gian khởi động hệ thống cuối cùng.

-r,--runlevel

Hiển thị mức chạy hiện tại.

-H,--heading

In tiêu đề cột.

So với `who`, `w` sẽ cho kết quả chi tiết hơn một chút:

```
root@debian:~# w
17:56:12 up 40 min, 2 users, load average: 0.04, 0.12, 0.09
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
carol pts/0 192.168.1.4 17:16 1.00s 0.15s 0.05s sshd: carol [priv]
mimi pts/1 192.168.1.4 17:28 15:08 0.05s 0.05s -bash
```

Dòng trên cùng cung cấp cho chúng ta thông tin về thời gian hiện tại (17:56:12), hệ thống đã hoạt động được bao lâu (up 40 min), số lượng người dùng hiện đang đăng nhập (2 users) và các số tải trọng trung bình (load average: 0.04, 0.12, 0.09). Các giá trị này đề cập đến số lượng

công việc trong hàng đợi chạy được tính trung bình trong 1, 5 và 15 phút vừa qua.

Sau đó, chúng ta có thể thấy tám cột. Hãy cùng xem chi tiết:

USER

Tên đăng nhập của người dùng.

TTY

Tên cửa sổ dòng lệnh mà người dùng đang sử dụng.

FROM

Máy chủ từ xa mà người dùng đã đăng nhập.

LOGIN@

Thời gian đăng nhập.

IDLE

Thời gian rảnh.

JCPU

Thời gian được sử dụng bởi tất cả các tiến trình được đính kèm với tty (bao gồm cả các công việc hiện đang chạy ngầm).

PCPU

Thời gian được sử dụng bởi tiến trình hiện tại (tiến trình đang hiển thị cho dòng lệnh WHAT).

WHAT

Dòng lệnh của tiến trình hiện tại.

Giống như với who, chúng ta cũng có thể truyền tên người dùng cho w:

```
root@debian:~# w mimi
18:23:15 up 1:07, 2 users, load average: 0.00, 0.02, 0.05
USER      TTY      FROM          LOGIN@      IDLE        JCPU        PCPU WHAT
mimi     pts/1    192.168.1.4   17:28       9:23       0.06s      0.06s -bash
```

Cấu hình và cách sử dụng sudo cơ bản

Như đã được nhắc tới ở trên, su cho phép người dùng chuyển sang bất kỳ một người dùng nào khác trong hệ thống, miễn là mật khẩu của người dùng mục tiêu được cung cấp. Đối với siêu người

dùng, việc để mật khẩu của nó được phân phối hoặc được (nhiều) người dùng biết sẽ khiến hệ thống gặp rủi ro và là một thói quen bảo mật rất tồi. Cách sử dụng cơ bản của `su` là `su - target-username`. Tuy nhiên, khi thay đổi thành siêu người dùng — tên người dùng đích có thể có hoặc không:

```
carol@debian:~$ su - root
Password:
root@debian:~# exit
logout
carol@debian:~$ su -
Password:
root@debian:~#
```

Việc sử dụng dấu gạch ngang (-) sẽ đảm bảo rằng môi trường của người dùng đích được tải. Không có nó, môi trường của người dùng cũ sẽ được giữ nguyên:

```
carol@debian:~$ su
Password:
root@debian:/home/carol#
```

Mặt khác, chúng ta cũng có lệnh `sudo`. Với nó, ta có thể thực thi lệnh với tư cách là siêu người dùng hoặc bất cứ một người dùng nào khác. Từ góc độ bảo mật, `sudo` là một lựa chọn tốt hơn nhiều so với `su` vì nó có hai ưu điểm chính:

1. để chạy lệnh với quyền gốc, người dùng không cần mật khẩu của siêu người dùng mà chỉ cần mật khẩu của người dùng đang gọi lệnh tuân thủ theo một chính sách bảo mật. Chính sách bảo mật mặc định là `sudoers` được chỉ định trong `/etc/sudoers` và `/etc/sudoers.d/*`.
2. `sudo` cho phép người dùng chạy các lệnh đơn lẻ với các đặc quyền nâng cao thay vì khởi chạy một vỏ con hoàn toàn mới cho siêu người dùng như `su`.

Cách sử dụng cơ bản của `sudo` là `sudo -u target-username command`. Tuy nhiên, để chạy lệnh dưới tên siêu người dùng, tùy chọn `-u target-username` là không cần thiết:

```
carol@debian:~$ sudo -u mimi whoami
mimi
carol@debian:~$ sudo whoami
root
```

NOTE

`sudoers` sẽ sử dụng một dấu thời gian cho mỗi một người dùng (và mỗi cửa sổ dòng lệnh) nhằm lưu trữ vào bộ đệm thông tin xác thực để người dùng có thể sử

dụng `sudo` mà không cần mật khẩu trong khoảng thời gian mặc định là mười lăm phút. Chúng ta có thể sửa đổi giá trị mặc định này bằng cách thêm tùy chọn `timestamp_timeout` làm một cài đặt mặc định (Defaults) trong `/etc/sudoers` (ví dụ: Defaults timestamp_timeout=1 sẽ đặt thời gian chờ bộ nhớ đệm thông tin xác thực thành một phút).

Tệp `/etc/sudoers`

Tệp cấu hình chính của `sudo` là `/etc/sudoers` (ngoài ra còn có thư mục `/etc/sudoers.d`). Đây là nơi xác định đặc quyền `sudo` của người dùng. Nói cách khác, chúng ta sẽ chỉ định ai có thể chạy lệnh nào cũng như người dùng nào trên máy nào — cũng như các cài đặt khác ở đây. Cú pháp được sử dụng sẽ như sau:

```
carol@debian:~$ sudo less /etc/sudoers
(...)
# User privilege specification
root    ALL=(ALL:ALL) ALL

# Allow members of group sudo to execute any command
%sudo  ALL=(ALL:ALL) ALL
(...)
```

Đặc tả đặc quyền cho siêu người dùng là `ALL=(ALL:ALL) ALL`. Đặc tả này được dịch là: siêu người dùng (`root`) có thể đăng nhập từ tất cả các máy chủ (`ALL`) với tư cách là tất cả mọi người dùng và tất cả mọi nhóm (`(ALL:ALL)`) và chạy tất cả các lệnh (`ALL`). Điều này cũng đúng với các thành viên của nhóm `sudo` — hãy lưu ý cách xác định tên nhóm bằng một dấu phần trăm (%) đứng trước.

Do đó, để người dùng `carol` có thể kiểm tra trạng thái `apache2` từ bất kỳ một máy chủ nào với tư cách là bất kỳ một người dùng hoặc nhóm nào, chúng ta sẽ thêm dòng sau vào tệp `sudoers`:

```
carol  ALL=(ALL:ALL) /usr/bin/systemctl status apache2
```

Chúng ta có thể sẽ muốn giúp `carol` tránh khỏi sự bất tiện khi phải cung cấp mật khẩu để chạy lệnh `systemctl status apache2`. Để làm được điều này, ta sẽ phải sửa đổi dòng đặc tả như sau:

```
carol  ALL=(ALL:ALL) NOPASSWD: /usr/bin/systemctl status apache2
```

Giả sử bây giờ chúng ta muốn giới hạn máy chủ của mình ở `192.168.1.7` và cho phép `carol` chạy `systemctl status apache2` với tư cách là người dùng `mimi`. Chúng ta sẽ sửa đổi dòng đặc tả

như sau:

```
carol 192.168.1.7=(mimi) /usr/bin/systemctl status apache2
```

Bây giờ, ta có thể kiểm tra trạng thái của máy chủ web Apache với tư cách là người dùng `mimi`:

```
carol@debian:~$ sudo -u mimi systemctl status apache2
• apache2.service - The Apache HTTP Server
  Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
  Active: active (running) since Tue 2020-06-09 13:12:19 CEST; 29min ago
  (...)
```

Nếu `carol` được thăng chức lên quản trị viên hệ thống và được trao tất cả các đặc quyền, cách tiếp cận dễ dàng nhất là đưa cô ấy vào nhóm `sudo` đặc biệt với `usermod` và tùy chọn `-G` (chúng ta cũng có thể sử dụng tùy chọn `-a` để đảm bảo rằng người dùng không bị xóa khỏi bất kỳ một nhóm nào khác mà họ có thể thuộc về):

```
root@debian:~# sudo useradd -aG sudo carol
```

NOTE

Trong họ bản phân phối Red Hat, nhóm `wheel` chính là phó bản của nhóm quản trị đặc biệt `sudo` trong hệ thống Debian.

Thay vì chỉnh sửa `/etc/sudoers` trực tiếp, ta chỉ cần sử dụng lệnh `visudo` dưới tên siêu người dùng (ví dụ như `visudo`). Lệnh này sẽ mở `/etc/sudoers` bằng trình soạn thảo văn bản được xác định trước của người dùng. Để thay đổi trình soạn thảo văn bản mặc định, ta có thể thêm tùy chọn `editor` làm một cài đặt Defaults (Mặc định) trong `/etc/sudoers`. Ví dụ: để thay đổi trình soạn thảo thành `nano`, ta sẽ thêm dòng sau:

```
Defaults editor=/usr/bin/nano
```

NOTE

Ngoài ra, chúng ta có thể chỉ định một trình soạn thảo văn bản thông qua biến môi trường `EDITOR` khi sử dụng `visudo` (ví dụ: `EDITOR=/usr/bin/nano visudo`).

Ngoài người dùng và nhóm, chúng ta cũng có thể sử dụng bí danh trong `/etc/sudoers`. Có ba loại bí danh chính mà ta có thể xác định: *bí danh máy chủ* (`Host_Alias`), *bí danh người dùng* (`User_Alias`) và *bí danh của lệnh* (`Cmdn_Alias`). Sau đây là một ví dụ:

```
# Host alias specification
```

```
Host_Alias SERVERS = 192.168.1.7, server1, server2

# User alias specification

User_Alias REGULAR_USERS = john, mary, alex

User_Alias PRIVILEGED_USERS = mimi

User_Alias ADMINS = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS

# Cmnd alias specification

Cmnd_Alias SERVICES = /usr/bin/systemctl *

# User privilege specification
root    ALL=(ALL:ALL) ALL
ADMINS  SERVERS=SERVICES

# Allow members of group sudo to execute any command
%sudo  ALL=(ALL:ALL) ALL
```

Xét tệp `sudoers` mẫu này, hãy cùng giải thích ba loại bí danh một cách chi tiết hơn:

Bí danh máy chủ

Chúng bao gồm một danh sách tên máy chủ, địa chỉ IP cũng như các mạng và nhóm mạng (với ký tự `+` đứng đằng trước) được phân tách bằng dấu phẩy. Mặt nạ mạng cũng có thể được chỉ định. Bí danh máy chủ `SERVERS` bao gồm một địa chỉ IP và hai tên máy chủ:

```
Host_Alias SERVERS = 192.168.1.7, server1, server2
```

Bí danh người dùng

Chúng bao gồm một danh sách các người dùng được phân tách bằng dấu phẩy được chỉ định làm tên người dùng, nhóm (với ký tự `%` đứng đằng trước) và nhóm mạng (với ký tự `+` đứng đằng trước). Chúng ta có thể loại trừ những người dùng cụ thể bằng ký hiệu `!`. Ví dụ: bí danh người dùng `ADMINS` bao gồm người dùng `carol`, các thành viên của nhóm `sudo` và những thành viên của bí danh người dùng `PRIVILEGE_USERS` không thuộc bí danh người dùng `REGULAR_USERS`:

```
User_Alias ADMINS = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS
```

Bí danh của lệnh

Chúng bao gồm một danh sách các lệnh và thư mục được phân tách bằng dấu phẩy. Nếu một thư mục được chỉ định, mọi tệp trong thư mục đó sẽ được bao gồm — tuy nhiên, các thư mục con sẽ bị bỏ qua. Bí danh lệnh SERVICES bao gồm một lệnh duy nhất với tất cả các lệnh con của nó — như được chỉ định bởi dấu hoa thị (*):

```
Cmnd_Alias SERVICES = /usr/bin/systemctl *
```

Do có chỉ định bí danh, dòng `ADMINS SERVERS=SERVICES` ở dưới phần `User privilege specification` (Đặc tả đặc quyền người dùng) có thể được dịch là: tất cả người dùng thuộc `ADMINS` có thể sử dụng `sudo` để chạy bất kỳ một lệnh nào trong `SERVICES` trên bất kỳ một máy chủ nào trong `SERVER`.

NOTE

Có một loại bí danh thứ tư mà ta có thể đưa vào `/etc/sudoers`: *bí danh đích danh* (`Runas_Alias`). Chúng rất giống với bí danh người dùng nhưng sẽ cho phép ta chỉ định người dùng theo *ID người dùng* (UID) của họ. Tính năng này có thể sẽ rất thuận tiện trong một số trường hợp nhất định.

Bài tập Hướng dẫn

1. Hãy hoàn thành bảng sau đây về các quyền đặc biệt:

Quyền đặc biệt	Biểu diễn số	Biểu diễn Biểu tượng	Tìm các tệp có <i>duy nhất bộ đặc Quyền này</i>
SUID			
SGID			

2. Việc hiển thị các tệp *chỉ có* bộ bit SUID hoặc SGID thường sẽ không mấy thực tế. Hãy thực hiện các tác vụ sau để chứng minh rằng các tìm kiếm của bạn có thể hiệu quả hơn:

- Tìm tất cả các tệp có SUID (và các quyền khác) được đặt trong `/usr/bin`:

- Tìm tất cả các tệp có SGID (và các quyền khác) được đặt trong `/usr/bin`:

- Tìm tất cả các tệp có SUID hoặc SGID được đặt trong `/usr/bin`:

3. `chage` cho phép bạn thay đổi thông tin hết hạn mật khẩu của người dùng. Với quyền gốc, hãy hoàn thành bảng sau bằng cách cung cấp các lệnh chính xác cho người dùng `mary`:

Ý nghĩa	Các lệnh <code>chage</code>
Tạo mật khẩu có hiệu lực trong 365 ngày.	
Yêu cầu người dùng thay đổi mật khẩu vào lần đăng nhập tiếp theo.	
Đặt số ngày tối thiểu giữa các lần thay đổi mật khẩu thành 1.	
Vô hiệu hóa việc hết hạn mật khẩu.	
Cho phép người dùng thay đổi mật khẩu của mình bất cứ lúc nào.	
Đặt thời gian cảnh báo thành 7 ngày và ngày hết hạn tài khoản đến ngày 20 tháng 8 năm 2050.	

Ý nghĩa	Các lệnh chage
In thông tin hết hạn mật khẩu hiện tại của người dùng.	

4. Hãy hoàn thành bảng sau với tiện ích mạng thích hợp:

Hành động	(Các) Lệnh
Hiển thị các tệp mạng cho máy chủ 192.168.1.55 trên cổng 22 bằng cách sử dụng lsof.	
Hiển thị các tiến trình truy cập cổng mặc định của máy chủ web Apache trên máy của bạn bằng fuser.	
Liệt kê tất cả các ổ nối udp đang nghe trên máy của bạn bằng cách sử dụng netstat.	
Quét các cổng 80 đến 443 trên máy chủ 192.168.1.55 bằng cách sử dụng nmap.	

5. Hãy thực hiện các tác vụ sau liên quan đến kích thước cài đặt thường trú (RSS) và ulimit với tư cách là người dùng thông thường:

- Hiển thị giới hạn mềm trên RSS tối đa:

- Hiển thị giới hạn cứng trên RSS tối đa:

- Đặt giới hạn mềm trên RSS tối đa thành 5.000 kilobyte:

- Đặt giới hạn cứng trên RSS tối đa thành 10.000 kilobyte:

- Cuối cùng, hãy thử tăng giới hạn cứng trên RSS tối đa lên đến 15.000 kilobyte. Bạn có thể làm được điều này không? Tại sao?

6. Hãy xem xét dòng đầu ra của lệnh last sau đây và trả lời các câu hỏi:

```
carol pts/0 192.168.1.4 Sun May 31 14:16 - 14:22 (00:06)
```

- carol có được kết nối từ máy chủ từ xa không? Tại sao?
- Phiên của carol đã kéo dài trong bao lâu?
- carol có được kết nối thông qua một cửa sổ dòng lệnh thuần văn bản cổ điển thực thụ không? Tại sao?

7. Hãy xem đoạn trích sau từ `/etc/sudoers` và trả lời câu hỏi bên dưới.

```
# Host alias specification

Host_Alias SERVERS = 192.168.1.7, server1, server2

# User alias specification

User_Alias REGULAR_USERS = john, mary, alex

User_Alias PRIVILEGED_USERS = mimi

User_Alias ADMINS = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS

# Cmnd alias specification

Cmnd_Alias WEB_SERVER_STATUS = /usr/bin/systemctl status apache2

# User privilege specification
root    ALL=(ALL:ALL) ALL
ADMINS  SERVERS=WEB_SERVER_STATUS

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
```

alex có thể kiểm tra trạng thái của Máy chủ Web Apache trên bất kỳ một máy chủ nào không? Tại sao?

Bài tập Mở rộng

1. Ngoài SUID và SGID, chúng ta còn có một quyền đặc biệt thứ ba là *bit dính* (sticky bit). Hiện tại, nó chủ yếu được sử dụng trên các thư mục như `/tmp` để ngăn người dùng thông thường xóa hoặc di chuyển các tệp khác ngoài tệp của họ. Hãy thực hiện các nhiệm vụ sau:

- Đặt *bit dính* trên `~/temporal`:

- Tìm các thư mục có *bit dính* (và bất kỳ quyền nào khác) được đặt trên thư mục chính của bạn:

- Bỏ đặt *bit dính* trên `~/temporal`:

2. Làm thế nào để có thể biết được mật khẩu của người dùng đã bị khóa thông qua `passwd -l username` hoặc `usermod -L username` bằng cách xem tệp `/etc/shadow`?

3. Lệnh `usermod` tương ứng với `chage -E date username` hoặc `chage --expiredate date username` là gì?

4. Hãy cho biết hai lệnh `nmap` khác nhau để quét toàn bộ 65535 cổng trên localhost:

Tóm tắt

Trong bài học này, chúng ta đã học cách thực hiện một số nhiệm vụ quản trị bảo mật. Các chủ đề sau đây đã được đề cập tới:

- Tìm các tệp có bộ quyền SUID và SGID đặc biệt.
- Đặt và thay đổi mật khẩu người dùng và xử lý thông tin lão hoá của mật khẩu.
- Sử dụng một số tiện ích mạng để khám phá các cổng mở trên máy chủ/mạng.
- Thiết lập giới hạn về tài nguyên hệ thống.
- Kiểm tra người dùng đã đăng nhập vào hệ thống hoặc hiện đang đăng nhập.
- Cách sử dụng và cấu hình sudo cơ bản (thông qua tệp /etc/sudoers).

Các lệnh và tệp đã được thảo luận trong bài học này:

find

Tìm kiếm tệp trong hệ thống phân cấp thư mục.

passwd

Thay đổi mật khẩu người dùng.

chmod

Thay đổi bit chế độ tệp.

chage

Thay đổi thông tin hết hạn của mật khẩu của người dùng.

lsof

Liệt kê các tệp đang mở.

fuser

Xác định các tiến trình sử dụng tệp hoặc ổ nối.

netstat

In các kết nối mạng.

nmap

Công cụ thăm dò mạng và quét cổng.

ulimit

Nhận và đặt giới hạn người dùng.

/etc/security/limits.conf

Tập cấu hình để áp dụng các hạn chế đối với người dùng.

last

In danh sách người dùng đăng nhập lần gần đây nhất.

lastb

In danh sách các lần đăng nhập không hợp lệ.

/var/log/wtmp

Cơ sở dữ liệu thông tin đăng nhập của người dùng.

who

Hiển thị ai đã đăng nhập.

w

Hiển thị ai đã đăng nhập và họ đang làm gì.

su

Thay đổi người dùng hoặc trở thành siêu người dùng.

sudo

Thực thi lệnh với tư cách một người dùng khác (bao gồm cả siêu người dùng).

/etc/sudoers

Tập cấu hình mặc định cho chính sách bảo mật `sudo`.

Đáp án Bài tập Hướng dẫn

1. Hãy hoàn thành bảng sau đây về các quyền đặc biệt:

Quyền đặc biệt	Biểu diễn số	Biểu diễn Biểu tượng	Tìm các tệp có <i>duy nhất bộ đặc Quyền này</i>
SUID	4000	s,S	find -perm 4000, find -perm u+s

Quyền đặc biệt	Biểu diễn số	Biểu diễn Biểu tượng	Tìm các tệp có <i>duy nhất bộ đặc Quyền này</i>
SGID	2000	s,S	find -perm 2000, find -perm g+s

2. Việc hiển thị các tệp *chỉ có* bộ bit SUID hoặc SGID thường sẽ không mấy thực tế. Hãy thực hiện các tác vụ sau để chứng minh rằng các tìm kiếm của bạn có thể hiệu quả hơn:

- Tìm tất cả các tệp có SUID (và các quyền khác) được đặt trong /usr/bin:

```
find /usr/bin -perm -4000 or find /usr/bin -perm -u+s
```

- Tìm tất cả các tệp có SGID (và các quyền khác) được đặt trong /usr/bin:

```
find /usr/bin -perm -2000 hoặc find /usr/bin -perm -g+s
```

- Tìm tất cả các tệp có SUID hoặc SGID được đặt trong /usr/bin:

```
find /usr/bin -perm /6000
```

3. chage cho phép bạn thay đổi thông tin hết hạn mật khẩu của người dùng. Với quyền gốc, hãy hoàn thành bảng sau bằng cách cung cấp các lệnh chính xác cho người dùng mary:

Ý nghĩa	Các lệnh chage
Tạo mật khẩu có hiệu lực trong 365 ngày.	chage -M 365 mary, chage --maxdays 365 mary
Yêu cầu người dùng thay đổi mật khẩu vào lần đăng nhập tiếp theo.	chage -d 0 mary, chage --lastday 0 mary
Đặt số ngày tối thiểu giữa các lần thay đổi mật khẩu thành 1.	chage -m 1 mary, chage --mindays 1 mary
Vô hiệu hóa việc hết hạn mật khẩu.	chage -M 99999 mary, chage --maxdays 99999 mary
Cho phép người dùng thay đổi mật khẩu của mình bất cứ lúc nào.	chage -m 0 mary, chage --mindays 0 mary
Đặt thời gian cảnh báo thành 7 ngày và ngày hết hạn tài khoản đến ngày 20 tháng 8 năm 2050.	chage -W 7 -E 2050-08-20 mary, chage --warndays 7 --expiredate 2050-08-20 mary

Ý nghĩa	Các lệnh chage
In thông tin hết hạn mật khẩu hiện tại của người dùng.	<code>chage -l mary, chage --list mary</code>

4. Hãy hoàn thành bảng sau với tiện ích mạng thích hợp:

Hành động	(Các) Lệnh
Hiển thị các tệp mạng cho máy chủ 192.168.1.55 trên cổng 22 bằng cách sử dụng <code>lsof</code> .	<code>lsof -i@192.168.1.55:22</code>
Hiển thị các tiến trình truy cập cổng mặc định của máy chủ web Apache trên máy của bạn bằng <code>fuser</code> .	<code>fuser -vn tcp 80, fuser --verbose --namespace tcp 80</code>
Liệt kê tất cả các ổ nối <code>udp</code> đang nghe trên máy của bạn bằng cách sử dụng <code>netstat</code> .	<code>netstat -lu, netstat --listening --udp</code>
Quét các cổng 80 đến 443 trên máy chủ 192.168.1.55 bằng cách sử dụng <code>nmap</code> .	<code>nmap -p 80-443 192.168.1.55</code>

5. Hãy thực hiện các tác vụ sau liên quan đến *kích thước cài đặt thường trú (RSS)* và `ulimit` với tư cách là người dùng thông thường:

- Hiển thị giới hạn *mềm* trên *RSS* tối đa:

```
ulimit -m, ulimit -Sm
```

- Hiển thị giới hạn *cứng* trên *RSS* tối đa:

```
ulimit -Hm
```

- Đặt giới hạn *mềm* trên *RSS* tối đa thành 5.000 kilobyte:

```
ulimit -Sm 5000
```

- Đặt giới hạn *cứng* trên *RSS* tối đa thành 10.000 kilobyte:

```
ulimit -Hm 10000
```

- Cuối cùng, hãy thử tăng giới hạn *cứng* trên *RSS* tối đa lên đến 15.000 kilobyte. Bạn có thể làm được điều này không? Tại sao?

Không. Một khi đã được đặt, người dùng thông thường không thể gia tăng giới hạn cứng.

6. Hãy xem xét dòng đầu ra của lệnh `last` sau đây và trả lời các câu hỏi:

```
carol pts/0 192.168.1.4 Sun May 31 14:16 - 14:22 (00:06)
```

- `carol` có được kết nối từ máy chủ từ xa không? Tại sao?

Có, địa chỉ IP của máy chủ từ xa nằm ở cột thứ ba.

- Phiên của `carol` đã kéo dài trong bao lâu?

Sáu phút (như được hiển thị ở cột cuối cùng).

- `carol` có được kết nối thông qua một cửa sổ dòng lệnh thuần văn bản cổ điển thực thụ không? Tại sao?

Không, `pts/0` trong cột thứ hai cho biết kết nối được thực hiện thông qua trình mô phỏng cửa sổ dòng lệnh đồ họa (hay còn gọi là *Pseudo Terminal Slave*).

7. Hãy xem đoạn trích sau từ `/etc/sudoers` và trả lời câu hỏi bên dưới.

```
# Host alias specification

Host_Alias SERVERS = 192.168.1.7, server1, server2

# User alias specification

User_Alias REGULAR_USERS = john, mary, alex

User_Alias PRIVILEGED_USERS = mimi

User_Alias ADMINS = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS

# Cmnd alias specification

Cmnd_Alias WEB_SERVER_STATUS = /usr/bin/systemctl status apache2

# User privilege specification
root    ALL=(ALL:ALL) ALL
ADMINS  SERVERS=WEB_SERVER_STATUS

# Allow members of group sudo to execute any command
%sudo  ALL=(ALL:ALL) ALL
```


alex có thể kiểm tra trạng thái của Máy chủ Web Apache trên bất kỳ một máy chủ nào không?
Tại sao?

No, as he is a member of `REGULAR_USERS` and that group of users is excluded from `ADMINS`; the only users (apart from `carol`, members of the `sudo` group and `root`) that can run `systemctl status apache2` on the `SERVERS`.

Đáp án Bài tập Mở rộng

1. Ngoài SUID và SGID, chúng ta còn có một quyền đặc biệt thứ ba là *bit dính* (sticky bit). Hiện tại, nó chủ yếu được sử dụng trên các thư mục như `/tmp` để ngăn người dùng thông thường xóa hoặc di chuyển các tệp khác ngoài tệp của họ. Hãy thực hiện các nhiệm vụ sau:

- Đặt *bit dính* trên `~/temporal`:

```
chmod +t temporal, chmod 1755 temporal
```

- Tìm các thư mục có *bit dính* (và bất kỳ quyền nào khác) được đặt trên thư mục chính của bạn:

```
find ~ -perm -1000, find ~ -perm /1000
```

- Bỏ đặt *bit dính* trên `~/temporal`:

```
chmod -t temporal, chmod 0755 temporal
```

2. Làm thế nào để có thể biết được mật khẩu của người dùng đã bị khóa thông qua `passwd -l username` hoặc `usermod -L username` bằng cách xem tệp `/etc/shadow`?

Dấu chấm than sẽ xuất hiện ở trường thứ hai ngay sau tên đăng nhập của người dùng bị khóa (ví dụ: `mary: !6g0g9xJgv...`).

3. Lệnh `usermod` tương ứng với `chage -E date username` hoặc `chage --expiredate date username` là gì?

```
usermod -e date username và usermod --expiredate date username
```

4. Hãy cho biết hai lệnh `nmap` khác nhau để quét toàn bộ 65535 cổng trên localhost:

```
nmap -p 1-65535 localhost và nmap -p- localhost
```



110.2 Thiết lập Bảo mật Máy chủ

Tham khảo các mục tiêu LPI

[LPIC-1 version 5.0, Exam 102, Objective 110.2](#)

Khối lượng

3

Các lĩnh vực kiến thức chính

- Kiến thức về mật khẩu ẩn và cách chúng hoạt động.
- Tắt các dịch vụ mạng không sử dụng.
- Hiểu vai trò của trình bao bọc TCP.

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- `/etc/nologin`
- `/etc/passwd`
- `/etc/shadow`
- `/etc/xinetd.d/`
- `/etc/xinetd.conf`
- `systemd.socket`
- `/etc/inittab`
- `/etc/init.d/`
- `/etc/hosts.allow`
- `/etc/hosts.deny`



110.2 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	110 Bảo mật
Mục tiêu:	110.2 Thiết lập Bảo mật Máy chủ
Bài:	1 trên 1

Giới thiệu

Chương này sẽ trình bày về bốn cách cơ bản để cải thiện bảo mật máy chủ:

1. Một số lệnh và cài đặt cấu hình cơ bản nhằm nâng cao tính bảo mật xác thực bằng mật khẩu ẩn.
2. Cách sử dụng các siêu trình nền để nghe các kết nối mạng đến.
3. Kiểm tra các dịch vụ mạng để tìm các trình nền không cần thiết.
4. Trình bao bọc TCP (TCP wrappers) như một loại tường lửa đơn giản.

Cải thiện Bảo mật xác thực bằng Mật khẩu ẩn

Các thành phần cơ bản của dữ liệu tài khoản người dùng được lưu trữ trong tệp `/etc/passwd`. Tệp này chứa bảy trường: tên đăng nhập, ID người dùng, ID nhóm, mật khẩu, chú thích (hay còn gọi là GECOS), vị trí thư mục chính và cuối cùng là vỏ mặc định. Một cách đơn giản để ghi nhớ thứ tự của các trường này là chia quá trình đăng nhập của người dùng ra làm các phần: đầu tiên là người dùng nhập tên đăng nhập; thứ hai là hệ thống ánh xạ tên đăng nhập này thành một ID người dùng (uid) và thứ ba là thành một ID nhóm (gid); thứ tư là hệ thống yêu cầu mật khẩu; thứ

năm là tra cứu chú thích; thứ sáu là vào thư mục chính của người dùng; thứ bảy là đặt vỏ mặc định.

Trong các hệ thống hiện đại, mật khẩu đã không còn được lưu trong tệp `/etc/passwd`. Thay vào đó, trường mật khẩu sẽ chỉ chứa một ký tự x viết thường. Tệp `/etc/passwd` phải đọc được bởi mọi người dùng nên chúng ta không nên lưu trữ mật khẩu ở đây. Thay vào đó, x biểu thị việc mật khẩu được mã hóa (băm) trên thực tế đang được lưu trữ trong tệp `/etc/shadow`. Tệp này không thể được đọc bởi mọi người dùng.

Các cài đặt mật khẩu được định cấu hình bằng lệnh `passwd` và `chage`. Cả hai lệnh đều sẽ thay đổi mục nhập của người dùng `emma` trong tệp `/etc/shadow`. Là một siêu người dùng, chúng ta có thể đặt mật khẩu cho người dùng `emma` bằng lệnh sau:

```
$ sudo passwd emma
New password:
Retype new password:
passwd: password updated successfully
```

Sau đó, chúng ta sẽ được nhắc hai lần để xác nhận mật khẩu mới.

Để liệt kê thời gian hết hạn mật khẩu và các cài đặt hết hạn mật khẩu khác cho người dùng `emma`, hãy sử dụng:

```
$ sudo chage -l emma
Last password change           : Apr 27, 2020
Password expires                : never
Password inactive              : never
Account expires                : never
Minimum number of days between password change : 0
Maximum number of days between password change : 99999
Number of days of warning before password expires : 7
```

Để ngăn người dùng `emma` đăng nhập vào hệ thống, siêu người dùng có thể đặt ngày hết hạn mật khẩu trước ngày hiện tại. Ví dụ: nếu ngày hôm nay là 27/03/2020, ta có thể đặt ngày hết hạn mật khẩu của người dùng bằng cách sử dụng một ngày trước đó:

```
$ sudo chage -E 2020-03-26 emma
```

Ngoài ra, siêu người dùng cũng có thể sử dụng:

```
$ sudo passwd -l emma
```

để khóa tài khoản tạm thời thông qua tùy chọn `-l` cho `passwd`. Để kiểm tra tác động của những thay đổi này, hãy thử đăng nhập bằng tài khoản `emma`:

```
$ sudo login emma
Password:
Your account has expired; please contact your system administrator

Authentication failure
```

Để ngăn tất cả mọi người dùng ngoại trừ siêu người dùng đăng nhập tạm thời vào hệ thống, siêu người dùng có thể tạo một tệp có tên là `/etc/nologin`. Tệp này có thể chứa thông báo gửi tới người dùng một thông báo về lý do tại sao họ không thể đăng nhập (ví dụ như thông báo bảo trì hệ thống). Để biết thêm chi tiết, hãy xem man 5 `nologin`. Hãy lưu ý rằng chúng ta còn có lệnh `nologin` có thể được sử dụng để ngăn một phiên đăng nhập khi được đặt làm vô mặc định cho người dùng. Ví dụ:

```
$ sudo usermod -s /sbin/nologin emma
```

Hãy xem man 8 `nologin` để biết thêm chi tiết.

Cách sử dụng Siêu Trình nền để nghe các Kết nối Mạng đến.

Các dịch vụ mạng như máy chủ web, máy chủ email và máy chủ in thường vận hành như một dịch vụ độc lập và lắng nghe trên một cổng mạng chuyên dụng. Tất cả các dịch vụ độc lập này đều chạy song song với nhau. Trên hệ thống dựa trên Sys-V-init cổ điển, mỗi dịch vụ này đều có thể được điều khiển bằng lệnh `service`. Trên các hệ thống dựa trên `systemd` hiện tại, ta có thể sử dụng `systemctl` để quản lý chúng.

Trước đây, nguồn tài nguyên máy tính sẵn có ít hơn rất nhiều và việc chạy song song nhiều dịch vụ ở chế độ độc lập không phải là một lựa chọn tốt. Thay vào đó, một siêu trình nền đã được sử dụng để lắng nghe các kết nối mạng đến và khởi động dịch vụ thích hợp theo yêu cầu. Phương pháp xây dựng kết nối mạng này mất nhiều thời gian hơn so với phương thức cũ. Các siêu trình nền phổ biến là `inetd` và `xinetd`. Trên các hệ thống hiện tại dựa trên `systemd`, đơn vị `systemd.socket` có thể được sử dụng theo cách tương tự. Trong phần này, chúng ta sẽ sử dụng `xinetd` để chặn các kết nối đến trình nền `sshd` và khởi động trình nền này theo yêu cầu để minh họa cách sử dụng siêu trình nền.

Chúng ta cần có vài sự chuẩn bị trước khi định cấu hình cho dịch vụ `xinetd`, . Việc sử dụng hệ thống dựa trên Debian hay Red Hat sẽ không thành vấn đề. Mặc dù những lý giải này được thử nghiệm với Debian/GNU Linux 9.9 nhưng chúng sẽ hoạt động trên mọi hệ thống Linux hiện tại có tính năng `systemd` mà không có bất kỳ thay đổi đáng kể nào. Trước tiên, hãy đảm bảo rằng các gói `openssh-server` và `xinetd` đã được cài đặt. Sau đó, hãy xác minh rằng dịch vụ SSH đang hoạt động với:

```
$ systemctl status sshd
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2020-04-27 09:33:48 EDT; 3h 11min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
  Process: 430 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
 Main PID: 460 (sshd)
    Tasks: 1 (limit: 1119)
   Memory: 5.3M
   CGroup: /system.slice/ssh.service
           └─460 /usr/sbin/sshd -D
```

Đồng thời kiểm tra xem dịch vụ SSH có đang lắng nghe trên cổng mạng tiêu chuẩn 22 hay không:

```
# lsof -i :22
COMMAND PID USER  FD  TYPE  DEVICE  SIZE/OFF  NODE NAME
sshd    1194 root   3u  IPv4 16053268      0t0  TCP *:ssh (LISTEN)
sshd    1194 root   4u  IPv6 16053270      0t0  TCP *:ssh (LISTEN)
```

Cuối cùng là dừng dịch vụ SSH bằng:

```
$ sudo systemctl stop sshd.service
```

Trong trường hợp cần thực hiện thay đổi này vĩnh viễn và tiếp tục suy trì sau khi khởi động lại hệ thống, hãy sử dụng `systemctl disable sshd.service`.

Bây giờ chúng ta có thể tạo tệp cấu hình `xinetd /etc/xinetd.d/ssh` với một số cài đặt cơ bản:

```
service ssh
{
    disable      = no
    socket_type = stream
```

```

protocol    = tcp
wait        = no
user        = root
server      = /usr/sbin/sshd
server_args = -i
flags       = IPv4
interface   = 192.168.178.1
}

```

Hãy khởi động lại dịch vụ xinetd bằng:

```
$ sudo systemctl restart xinetd.service
```

Hãy kiểm tra xem dịch vụ nào hiện đang lắng nghe các kết nối SSH đến.

```
$ sudo lsof -i :22
COMMAND  PID USER  FD  TYPE  DEVICE  SIZE/OFF  NODE  NAME
xinetd   24098 root   5u  IPv4  7345141      0t0  TCP  192.168.178.1:ssh (LISTEN)
```

Chúng ta có thể thấy rằng dịch vụ xinetd đã nắm quyền kiểm soát truy cập vào cổng 22.

Dưới đây là một số chi tiết khác về cấu hình xinetd. Tập cấu hình chính là `/etc/xinetd.conf`:

```

# Simple configuration file for xinetd
#
# Some defaults, and include /etc/xinetd.d/

defaults
{
    # Please note that you need a log_type line to be able to use log_on_success
    # and log_on_failure. The default is the following :
    # log_type = SYSLOG daemon info
}

includedir /etc/xinetd.d

```

Ngoài các cài đặt mặc định, chúng ta chỉ có một chỉ dẫn để đặt một thư mục bao hàm (include directory). Trong thư mục này, ta có thể thiết lập một tập cấu hình duy nhất cho mỗi dịch vụ cần

xinetd xử lý. Chúng ta đã thực hiện điều này ở trên cho dịch vụ SSH và đặt tên tệp là `/etc/xinetd.d/ssh`. Tên của các tệp cấu hình có thể được chọn tùy ý ngoại trừ các tên tệp có chứa dấu chấm (.) hoặc kết thúc bằng dấu ngã (~). Tuy nhiên, thông lệ phổ biến là đặt tên tệp theo tên dịch vụ cần định cấu hình.

Một số tệp cấu hình trong thư mục `/etc/xinetd.d/` đã được bản phân phối cung cấp sẵn:

```
$ ls -l /etc/xinetd.d
total 52
-rw-r--r-- 1 root root 640 Feb  5  2018 chargen
-rw-r--r-- 1 root root 313 Feb  5  2018 chargen-udp
-rw-r--r-- 1 root root 502 Apr 11 10:18 daytime
-rw-r--r-- 1 root root 313 Feb  5  2018 daytime-udp
-rw-r--r-- 1 root root 391 Feb  5  2018 discard
-rw-r--r-- 1 root root 312 Feb  5  2018 discard-udp
-rw-r--r-- 1 root root 422 Feb  5  2018 echo
-rw-r--r-- 1 root root 304 Feb  5  2018 echo-udp
-rw-r--r-- 1 root root 312 Feb  5  2018 servers
-rw-r--r-- 1 root root 314 Feb  5  2018 services
-rw-r--r-- 1 root root 569 Feb  5  2018 time
-rw-r--r-- 1 root root 313 Feb  5  2018 time-udp
```

Các tệp này có thể được sử dụng làm bản mẫu trong những trường hợp hiếm hoi người dùng cần sử dụng một số dịch vụ cũ chẳng hạn như `daytime` (một triển khai rất cũ của máy chủ thời gian). Tất cả các tệp mẫu này đều có chứa chỉ thị `disable = Yes`.

Dưới đây là một số chi tiết khác về các lệnh được sử dụng trong tệp ví dụ `/etc/xinetd.d/ssh` cho ssh ở trên.

```
service ssh
{
    disable      = no
    socket_type  = stream
    protocol    = tcp
    wait        = no
    user        = root
    server      = /usr/sbin/sshd
    server_args = -i
    flags       = IPv4
    interface   = 192.168.178.1
}
```

service

Liệt kê các dịch vụ mà `xinetd` phải kiểm soát. Chúng ta có thể sử dụng số cổng (chẳng hạn như 22) hoặc tên được ánh xạ tới số cổng trong `/etc/services` (ví dụ như `ssh`).

```
{
```

Cài đặt chi tiết bắt đầu bằng dấu ngoặc nhọn mở.

disable

Để kích hoạt các cài đặt này, hãy đặt cài đặt này thành `no`. Nếu muốn vô hiệu hoá cài đặt tạm thời, hãy đặt nó thành `yes`.

socket_type

Ta có thể chọn `stream` cho ổ nối TCP hoặc `dgram` cho ổ nối UDP và nhiều hơn thế.

protocol

Chọn TCP hoặc UDP.

wai

Đối với các kết nối TCP, giá trị này thường được đặt thành `no`.

user

Dịch vụ bắt đầu ở dòng này sẽ thuộc sở hữu của người dùng này.

server

Đường dẫn đầy đủ đến dịch vụ sẽ được khởi động bởi `xinetd`.

server_args

Chúng ta có thể thêm tùy chọn cho dịch vụ tại đây. Nếu được khởi động bởi một siêu máy chủ, nhiều dịch vụ sẽ yêu cầu một tùy chọn đặc biệt. Đối với SSH, đây sẽ là tùy chọn `-i`.

flags

Ta có thể chọn IPv4, IPv6 và các loại khác.

interface

Giao diện mạng mà `xinetd` phải điều khiển. Lưu ý: ta cũng có thể chọn lệnh `bind` (đồng nghĩa với `interface`).

```
}
```

Kết thúc bằng một dấu ngoặc nhọn đóng.

Những dịch vụ kế thừa được khởi động bởi siêu máy chủ `xinetd` là các đơn vị ổ nối `systemd`. Việc

Thiết lập một đơn vị ổ nối systemd rất đơn giản và dễ dàng vì luôn có sẵn một đơn vị ổ nối systemd được xác định trước cho SSH. Hãy đảm bảo rằng dịch vụ `xinetd` và SSH đang không chạy.

Bây giờ, chúng ta chỉ cần khởi động ổ nối SSH:

```
$ sudo systemctl start ssh.socket
```

Để kiểm tra dịch vụ nào hiện đang nghe trên cổng 22, chúng ta sẽ sử dụng lại `lsof`. Hãy lưu ý tùy chọn `-P` ở đây đã được sử dụng để hiển thị số cổng thay vì tên dịch vụ ở đầu ra:

```
$ sudo lsof -i :22 -P
COMMAND PID USER  FD   TYPE    DEVICE  SIZE/OFF  NODE NAME
systemd   1  root   57u  IPv6  14730112      0t0  TCP *:22 (LISTEN)
```

Để hoàn tất phiên này, chúng ta nên thử đăng nhập vào máy chủ bằng một ứng dụng khách SSH tùy chọn.

TIP

Trong trường hợp `systemctl start ssh.socket` không hoạt động với bản phân phối, hãy thử `systemctl start sshd.socket`.

Kiểm tra các dịch vụ mạng để tìm các trình nền không cần thiết.

Vì lý do bảo mật cũng như để kiểm soát tài nguyên hệ thống, quản trị viên phải cần có một cái nhìn tổng quan về những dịch vụ đang chạy. Các dịch vụ không cần thiết và không được sử dụng nên bị vô hiệu hóa. Ví dụ: trong trường hợp người dùng không cần phân phối các trang web thì việc chạy máy chủ web như Apache hoặc nginx là không cần thiết.

Trên các hệ thống dựa trên Sys-V-init, chúng ta có thể kiểm tra trạng thái của tất cả các dịch vụ bằng cách sau:

```
$ sudo service --status-all
```

Hãy xác minh xem mỗi dịch vụ được liệt kê từ đầu ra của lệnh có cần thiết hay không và vô hiệu hóa tất cả các dịch vụ không cần thiết bằng (đối với các hệ thống dựa trên Debian):

```
$ sudo update-rc.d SERVICE-NAME remove
```

Đối với các hệ thống dựa trên Red Hat, chúng ta sẽ sử dụng:

```
$ sudo chkconfig SERVICE-NAME off
```

Trên các hệ thống dựa trên systemd hiện đại, chúng ta có thể sử dụng thông tin sau để liệt kê tất cả các dịch vụ đang chạy:

```
$ systemctl list-units --state active --type service
```

Sau đó, chúng ta có thể vô hiệu hóa từng đơn vị dịch vụ không cần thiết bằng:

```
$ sudo systemctl disable UNIT --now
```

Lệnh này sẽ dừng dịch vụ và xóa nó khỏi danh sách các dịch vụ để ngăn nó khởi động ở lần khởi động hệ thống tiếp theo.

Ngoài ra, để nhận một bản khảo sát về các dịch vụ mạng đang nghe, chúng ta có thể sử dụng `netstat` trên các hệ thống cũ hơn (miễn là hệ thống đã được cài đặt gói `net-tools`):

```
$ netstat -ltu
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN
tcp      0      0 localhost:mysql        0.0.0.0:*               LISTEN
tcp6     0      0 [::]:http              [::]:*                 LISTEN
tcp6     0      0 [::]:ssh                [::]:*                 LISTEN
udp      0      0 0.0.0.0:bootpc         0.0.0.0:*
```

Hoặc trên các hệ thống hiện đại, chúng ta có thể sử dụng lệnh tương đương là `ss` (viết tắt của "socket services"):

```
$ ss -ltu
Netid  State      Recv-Q    Send-Q    Local Address:Port      Peer
Address:Port
udp    UNCONN    0          0          0.0.0.0:bootpc
0.0.0.0:*
tcp    LISTEN    0          128       0.0.0.0:ssh
0.0.0.0:*
tcp    LISTEN    0          80        127.0.0.1:mysql
0.0.0.0:*
tcp    LISTEN    0          128              *:http
```

```
*:*
tcp      LISTEN   0        128      [::]:ssh
[::]:*
```

Trình bao bọc TCP như một loại tường lửa đơn giản.

Vào thời điểm Linux chưa có tường lửa, trình bao bọc TCP đã được sử dụng để bảo mật các kết nối mạng vào máy chủ. Ngày nay, nhiều chương trình không tuân theo trình bao bọc TCP nữa. Trong các bản phân phối dựa trên Red Hat (ví dụ như Fedora 29) gần đây, các hỗ trợ cho trình bao bọc TCP đã bị loại bỏ hoàn toàn. Để hỗ trợ các hệ thống Linux cũ vẫn sử dụng trình bao bọc TCP, quản trị viên cần có một số kiến thức cơ bản về công nghệ cụ thể này.

Chúng ta sẽ một lần nữa sử dụng dịch vụ SSH làm một ví dụ cơ bản. Dịch vụ trên máy chủ ví dụ của chúng ta chỉ có thể truy cập được từ mạng cục bộ. Trước tiên, chúng ta sẽ kiểm tra xem trình nền SSH có sử dụng thư viện `libwrap` cung cấp các hỗ trợ cho trình bao bọc TCP hay không:

```
$ ldd /usr/sbin/sshd | grep "libwrap"
libwrap.so.0 => /lib/x86_64-linux-gnu/libwrap.so.0 (0x00007f91dbec0000)
```

Sau đó, hãy thêm dòng sau vào tệp `/etc/hosts.deny`:

```
sshd: ALL
```

Cuối cùng, chúng ta sẽ định cấu hình một ngoại lệ trong tệp `/etc/hosts.allow` cho các kết nối SSH từ mạng cục bộ:

```
sshd: LOCAL
```

Những thay đổi sẽ có hiệu lực ngay lập tức mà không cần phải khởi động lại bất kỳ một dịch vụ nào. Chúng ta có thể kiểm tra điều này với ứng dụng khách `ssh`.

Bài tập Hướng dẫn

1. Làm cách nào để có thể mở khóa tài khoản emma đã bị khóa trước đó?

2. Trước đây tài khoản emma đã được đặt ngày hết hạn. Làm cách nào để đặt ngày hết hạn thành "không bao giờ" (never)?

3. Hãy tưởng tượng dịch vụ in CUPS đang xử lý các lệnh in không cần thiết trên máy chủ của bạn. Làm thế nào để có thể vô hiệu hóa dịch vụ vĩnh viễn? Làm thế nào để có thể kiểm tra cổng thích hợp không còn hoạt động nữa?

4. Bạn đã cài đặt máy chủ web nginx. Làm cách nào để có thể kiểm tra xem nginx có hỗ trợ trình bao bọc TCP hay không?

Bài tập Mở rộng

1. Hãy kiểm tra xem sự tồn tại của tệp `/etc/nologin` có ngăn cản việc đăng nhập của người dùng `root` hay không.

2. Sự tồn tại của tệp `/etc/nologin` có ngăn cản việc đăng nhập không cần mật khẩu bằng khóa SSH không?

3. Điều gì sẽ xảy ra trong quá trình đăng nhập khi tệp `/etc/nologin` chỉ chứa dòng văn bản này `login currently is not possible` (hiện tại không thể đăng nhập)?

4. Người dùng bình thường `emma` có thể lấy được thông tin về người dùng `root` có trong tệp `/etc/passwd` bằng lệnh `grep root /etc/passwd` không?

5. Người dùng bình thường `emma` có thể lấy thông tin về mật khẩu băm của chính cô ấy có trong tệp `/etc/shadow` bằng lệnh `grep emma /etc/shadow`?

6. Cần phải thực hiện những bước nào để kích hoạt và kiểm tra thời gian ban ngày cũ do `xinetd` xử lý? Hãy lưu ý đây chỉ là một bài tập mở rộng và đừng thực hiện việc này trong một môi trường thực tế.

Tóm tắt

Trong bài học này, chúng ta đã học về:

1. Tập nơi mật khẩu được lưu trữ cũng như một số cài đặt bảo mật mật khẩu (ví dụ như thời gian hết hạn).
2. Mục đích của siêu trình nền `xinetd` và cách chạy nó cũng như khởi động dịch vụ `sshd` theo yêu cầu.
3. Để kiểm tra những dịch vụ mạng nào đang chạy và cách vô hiệu hoá những dịch vụ không cần thiết.
4. Sử dụng Trình bao bọc TCP như một loại tường lửa đơn giản.

Các lệnh được sử dụng trong bài học này bao gồm:

chage

Thay đổi tuổi mật khẩu của người dùng.

chkconfig

Một lệnh cổ điển ban đầu được sử dụng trên các hệ thống dựa trên Red Hat để đặt xem một dịch vụ có nên được khởi động trong quá trình khởi động hệ thống hay không.

netstat

Một tiện ích cổ điển (hiện có trong gói `net-tools`) hiển thị các trình nền truy cập các cổng mạng trên hệ thống và cách sử dụng chúng.

nologin

Một lệnh có thể được sử dụng thay cho vỏ của người dùng để ngăn họ đăng nhập.

passwd

Được sử dụng để tạo hoặc thay đổi mật khẩu của người dùng.

service

Một phương pháp cũ hơn để kiểm soát trạng thái của trình nền (chẳng hạn như dừng hoặc bắt đầu một dịch vụ).

ss

Phó bản hiện đại của `netstat` nhưng cũng hiển thị thêm thông tin về các ổ nối khác nhau đang được sử dụng trên hệ thống.

systemctl

Lệnh điều khiển hệ thống được sử dụng để kiểm soát các khía cạnh khác nhau của các dịch vụ và ổ nối trên máy tính bằng systemd.

update-rc.d

Một lệnh cổ điển tương tự như `chkconfig` cho phép hoặc vô hiệu hóa một chương trình khởi động trong quá trình khởi động hệ thống trên các bản phân phối dựa trên Debian.

xinetd

Một siêu trình nền có thể kiểm soát quyền truy cập vào dịch vụ mạng theo yêu cầu, từ đó khiến dịch vụ không hoạt động cho đến khi nó thực sự được yêu cầu thực hiện một số tác vụ.

Đáp án Bài tập Hướng dẫn

1. Làm cách nào để có thể mở khóa tài khoản emma đã bị khóa trước đó?

Siêu người dùng có thể chạy `passwd -u emma` để mở khóa tài khoản.

2. Trước đây tài khoản emma đã được đặt ngày hết hạn. Làm cách nào để đặt ngày hết hạn thành "không bao giờ" (never)?

Siêu người dùng có thể sử dụng `chage -E -1 emma` để đặt ngày hết hạn thành không bao giờ. Cài đặt này có thể được kiểm tra bằng `chage -l emma`.

3. Hãy tưởng tượng dịch vụ in CUPS đang xử lý các lệnh in không cần thiết trên máy chủ của bạn. Làm thế nào để có thể vô hiệu hóa dịch vụ vĩnh viễn? Làm thế nào để có thể kiểm tra cổng thích hợp không còn hoạt động nữa?

Sử dụng lệnh với tư cách là siêu người dùng:

```
systemctl disable cups.service --now
```

Sau đó, bạn có thể kiểm tra với

```
netstat -l | grep ":ipp " hoặc `ss -l | grep ":ipp ".
```

4. Bạn đã cài đặt máy chủ web nginx. Làm cách nào để có thể kiểm tra xem nginx có hỗ trợ trình bao bọc TCP hay không?

Lệnh

```
ldd /usr/sbin/nginx | grep "libwrap"
```

sẽ hiển thị một mục nhập trong trường hợp nginx có hỗ trợ trình bao bọc TCP.

Đáp án Bài tập Mở rộng

1. Hãy kiểm tra xem sự tồn tại của tệp `/etc/nologin` có ngăn cản việc đăng nhập của người dùng `root` hay không.

Người dùng `root` vẫn có thể đăng nhập.

2. Sự tồn tại của tệp `/etc/nologin` có ngăn cản việc đăng nhập không cần mật khẩu bằng khóa SSH không?

Có, việc đăng nhập không cần mật khẩu cũng sẽ bị ngăn chặn.

3. Điều gì sẽ xảy ra trong quá trình đăng nhập khi tệp `/etc/nologin` chỉ chứa dòng văn bản này `login currently is not possible` (hiện tại không thể đăng nhập)?

Thông báo `login currently is not possible` sẽ được hiển thị và một phiên đăng nhập sẽ bị ngăn chặn.

4. Người dùng bình thường `emma` có thể lấy được thông tin về người dùng `root` có trong tệp `/etc/passwd` bằng lệnh `grep root /etc/passwd` không?

Có vì tất cả mọi người dùng đều có quyền đọc tệp này.

5. Người dùng bình thường `emma` có thể lấy thông tin về mật khẩu băm của chính cô ấy có trong tệp `/etc/shadow` bằng lệnh `grep emma /etc/shadow`?

Không vì người dùng thông thường không có quyền đọc tệp này.

6. Cần phải thực hiện những bước nào để kích hoạt và kiểm tra dịch vụ cũ do `xinetd` xử lý? Hãy lưu ý đây chỉ là một bài tập mở rộng và đừng thực hiện việc này trong một môi trường thực tế.

Đầu tiên, hãy thay đổi tệp `/etc/xinetd.d/daytime` và đặt lệnh `disable = no`. Tiếp theo, hãy khởi động lại dịch vụ `xinetd` `systemctl restart xinetd.service` (hoặc `service xinetd restart` trên các hệ thống có `SyS-V-Init`). Sau đó, bạn có thể kiểm tra xem nó có hoạt động không với `nc localhost daytime`. Thay vì `nc`, bạn cũng có thể sử dụng `netcat`.



110.3 Bảo mật Dữ liệu bằng Mã hóa

Tham khảo các mục tiêu LPI

LPIC-1 version 5.0, Exam 102, Objective 110.3

Khối lượng

4

Các lĩnh vực kiến thức chính

- Thực hiện cấu hình và sử dụng máy khách OpenSSH 2 cơ bản.
- Hiểu vai trò của khóa máy chủ OpenSSH 2.
- Thực hiện cấu hình, sử dụng và thu hồi GnuPG cơ bản.
- Sử dụng GPG để mã hóa, giải mã, ký và xác minh tệp.
- Hiểu về các đường hầm cổng SSH (bao gồm cả đường hầm X11).

Sau đây là danh sách một phần các tệp, thuật ngữ và tiện ích được sử dụng

- `ssh`
- `ssh-keygen`
- `ssh-agent`
- `ssh-add`
- `~/.ssh/id_rsa` và `id_rsa.pub`
- `~/.ssh/id_dsa` và `id_dsa.pub`
- `~/.ssh/id_ecdsa` và `id_ecdsa.pub`
- `~/.ssh/id_ed25519` và `id_ed25519.pub`
- `/etc/ssh/ssh_host_rsa_key` và `ssh_host_rsa_key.pub`

- `/etc/ssh/ssh_host_dsa_key` và `ssh_host_dsa_key.pub`
- `/etc/ssh/ssh_host_ecdsa_key` và `ssh_host_ecdsa_key.pub`
- `/etc/ssh/ssh_host_ed25519_key` và `ssh_host_ed25519_key.pub`
- `~/.ssh/authorized_keys`
- `ssh_known_hosts`
- `gpg`
- `gpg-agent`
- `~/.gnupg/`



110.3 Bài 1

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	110 Bảo mật
Mục tiêu:	110.3 Bảo mật Dữ liệu bằng Mã hóa
Bài:	1 trên 2

Giới thiệu

Bảo mật dữ liệu bằng mã hóa là một điều hết sức quan trọng trong nhiều khía cạnh của quản trị hệ thống ngày nay — thậm chí còn quan trọng hơn nữa trong trường hợp truy cập hệ thống từ xa. Ngược lại với các giải pháp không an toàn như *telnet*, *rlogin* hoặc *FTP*, giao thức *SSH (Secure SHell - Giao thức Vô an toàn)* được thiết kế chú trọng đến bảo mật. Sử dụng mật mã khóa công khai, SSH sẽ xác thực cả máy chủ và người dùng, đồng thời mã hóa tất cả các trao đổi về thông tin diễn ra sau đó. Hơn nữa, SSH có thể được sử dụng để thiết lập các *cổng hầm* — bên cạnh nhiều tiện ích thứ khác — cho phép giao thức không được mã hóa truyền dữ liệu qua kết nối SSH được mã hóa. Phiên bản hiện tại được đề xuất của giao thức SSH là 2.0. *OpenSSH* là một triển khai mã nguồn mở và tự do của giao thức SSH.

Bài học này sẽ trình bày về cấu hình máy khách *OpenSSH* cơ bản cũng như vai trò của các khóa máy chủ *OpenSSH*. Khái niệm về đường hầm cổng SSH cũng sẽ được đề cập tới. Chúng ta sẽ sử dụng hai máy với thiết lập sau:

Vai trò của máy	Hệ điều hành	Địa chỉ IP	Tên máy chủ	Người dùng
Máy khách	Debian GNU/Linux 10 (buster)	192.168.1.55	debian	carol
Máy chủ	openSUSE Leap 15.1	192.168.1.77	halof	ina

Cấu hình và cách sử dụng Máy khách OpenSSH cơ bản

Mặc dù máy chủ và máy khách OpenSSH có các gói riêng biệt nhưng thông thường, chúng ta có thể cài đặt một siêu gói sẽ cung cấp cả hai cùng một lúc. Để thiết lập một phiên từ xa với máy chủ SSH, ta sẽ sử dụng lệnh `ssh` và chỉ định người dùng mà ta cần kết nối trên máy từ xa và địa chỉ IP hoặc tên máy chủ của máy từ xa. Vào lần đầu tiên kết nối với máy chủ từ xa, chúng ta sẽ nhận được thông báo như sau:

```
carol@debian:~$ ssh ina@192.168.1.77
The authenticity of host '192.168.1.77 (192.168.1.77)' can't be established.
ECDSA key fingerprint is SHA256:5JF7anupYipByCQm2BPvDHRVFJJixeslmp2NwATYI.
Are you sure you want to continue connecting (yes/no)?
```

Sau khi gõ `yes` và nhấn Enter, chúng ta sẽ được hỏi về mật khẩu của người dùng từ xa. Nếu nhập thành công, thông báo cảnh báo sẽ xuất hiện và sau đó ta sẽ được đăng nhập vào máy chủ từ xa:

```
Warning: Permanently added '192.168.1.77' (ECDSA) to the list of known hosts.
Password:
Last login: Sat Jun 20 10:52:45 2020 from 192.168.1.4
Have a lot of fun...
ina@halof:~>
```

Các thông báo đều khá là dễ hiểu: vì đây là lần đầu tiên thiết lập kết nối với máy chủ từ xa 192.168.1.77 nên tính xác thực của nó không thể được kiểm tra đối chứng với bất kỳ một cơ sở dữ liệu nào. Do đó, máy chủ từ xa đã cung cấp một dấu vân tay khóa ECDSA (ECDSA key fingerprint) của khóa công khai của nó (sử dụng hàm băm SHA256). Sau khi chấp nhận kết nối, khóa công khai của máy chủ từ xa sẽ được thêm vào cơ sở dữ liệu *các máy chủ đã biết* (knownhosts), từ đó cho phép xác thực máy chủ cho các kết nối trong tương lai. Danh sách khóa công khai của *các máy chủ đã biết* này được lưu giữ trong tệp `known_hosts` nằm trong `~/.ssh`:

```
ina@halof:~> exit
```

```
logout
Connection to 192.168.1.77 closed.
carol@debian:~$ ls .ssh/
known_hosts
```

Cả `.ssh` và `known_hosts` đều đã được tạo sau khi kết nối từ xa đầu tiên được thiết lập. `~/ .ssh` là thư mục mặc định cho thông tin xác thực và cấu hình dành riêng cho người dùng.

NOTE Bạn cũng có thể sử dụng `ssh` để thực thi một lệnh duy nhất trên máy chủ từ xa và sau đó quay lại cửa sổ dòng lệnh cục bộ của mình (ví dụ: chạy `ssh ina@halof ls`).

Nếu đang sử dụng cùng một người dùng trên cả máy chủ cục bộ và máy chủ từ xa thì ta không cần chỉ định tên người dùng khi thiết lập kết nối SSH. Ví dụ: nếu đăng nhập với tư cách là người dùng `carol` trên `debian` và muốn kết nối với `halof` cũng với tư cách người dùng `carol`, ta chỉ cần gõ `ssh 192.168.1.77` hoặc `ssh halof` (nếu tên có thể được phân giải):

```
carol@debian:~$ ssh halof
Password:
Last login: Wed Jul 1 23:45:02 2020 from 192.168.1.55
Have a lot of fun...
carol@halof:~>
```

Bây giờ, giả sử chúng ta thiết lập một kết nối từ xa mới với một máy chủ có cùng địa chỉ IP như `halof` (một điều khá phổ biến nếu ta sử dụng DHCP trong mạng LAN của mình). Chúng ta sẽ được cảnh báo về khả năng xảy ra một cuộc tấn công từ *xen giữa* (man-in-the-middle):

```
carol@debian:~$ ssh john@192.168.1.77
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!    @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the ECDSA key sent by the remote host is
SHA256:KH4q3vP6C7e0SEjyG8Wlz9fV1f+jmWJ5139RBxBh3TY.
Please contact your system administrator.
Add correct host key in /home/carol/.ssh/known_hosts to get rid of this message.
Offending ECDSA key in /home/carol/.ssh/known_hosts:1
  remove with:
  ssh-keygen -f "/home/carol/.ssh/known_hosts" -R "192.168.1.77"
ECDSA host key for 192.168.1.77 has changed and you have requested strict checking.
```



```
Host key verification failed.
```

Vì không phải đối phó với một cuộc tấn công *xen giữa* nên chúng ta có thể thêm vân tay khóa công khai của máy chủ mới vào `.ssh/known_hosts` một cách an toàn. Như thông báo cho biết, trước tiên, chúng ta có thể sử dụng lệnh `ssh-keygen -f "/home/carol/.ssh/known_hosts" -R "192.168.1.77"` để xóa khóa *vi phạm* (một cách khác là sử dụng `ssh-keygen -R 192.168.1.77` để xóa tất cả các khóa thuộc về 192.168.1.77 khỏi `~/ .ssh/known_hosts`). Sau đó, chúng ta sẽ có thể thiết lập kết nối với máy chủ mới.

Đăng nhập dựa trên Khóa

Ta có thể thiết lập máy khách SSH của mình để không cần phải cung cấp bất kỳ một mật khẩu nào khi đăng nhập mà thay vào đó là sử dụng các khóa công khai. Đây là phương pháp kết nối được ưa thích với máy chủ từ xa thông qua SSH vì nó an toàn hơn rất nhiều. Điều đầu tiên chúng ta phải làm là tạo một cặp khóa trên máy khách. Để thực hiện việc này, ta sẽ sử dụng `ssh-keygen` với tùy chọn `-t` chỉ định loại mã hóa cần dùng (trong trường hợp của chúng ta là *Thuật toán chữ kí số đường cong Elip*, viết tắt là ECDSA). Sau đó, ta sẽ được yêu cầu về đường dẫn để lưu cặp khóa (`~/ .ssh/` và vị trí mặc định đều rất thuận tiện) và cụm mật khẩu. Mặc dù cụm mật khẩu là tùy chọn nhưng người dùng được khuyến nghị luôn luôn sử dụng nó.

```
carol@debian:~/ .ssh$ ssh-keygen -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/home/carol/.ssh/id_ecdsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/carol/.ssh/id_ecdsa.
Your public key has been saved in /home/carol/.ssh/id_ecdsa.pub.
The key fingerprint is:
SHA256:tlamD0SaTquPZYdNepwj8XN4xvqmHCbe8g5FKKUfMo8 carol@debian
The key's randomart image is:
+---[ECDSA 256]---+
|   .               |
|  o .              |
| = o o             |
|  B *              |
| E B S o           |
|  o & 0            |
|   @ ^ =           |
|  * .@ @.          |
|  o.o+B+o          |
+-----[SHA256]-----+
```

NOTE

Khi tạo cặp khóa, ta có thể truyền cho `ssh-keygen` tùy chọn `-b` để chỉ định kích thước khóa theo bit (ví dụ: `ssh-keygen -t ecdsa -b 521`).

Lệnh trên đã tạo thêm hai tệp nữa trong thư mục `~/ .ssh` của chúng ta:

```
carol@debian:~/ .ssh$ ls
id_ecdsa id_ecdsa.pub known_hosts
```

id_ecdsa

Đây là khóa riêng tư.

id_ecdsa.pub

Đây là khóa công khai.

NOTE

Trong mật mã bất đối xứng (còn gọi là mật mã khóa công khai), khóa công khai và khóa riêng tư có liên quan về mặt toán học với nhau trên phương diện là bất cứ thứ gì được mã hóa bởi cái này sẽ chỉ có thể được giải mã bởi cái kia.

Điều tiếp theo chúng ta cần làm là thêm khoá công khai của mình vào tệp `~/ .ssh/authorized_keys` của người dùng mà ta sẽ sử dụng để đăng nhập vào máy chủ từ xa (nếu thư mục `~/ .ssh` chưa tồn tại, chúng ta sẽ phải tạo nó trước). Ta có thể sao chép khóa công khai của mình vào máy chủ từ xa theo một số cách: sử dụng ổ flash USB, thông qua lệnh `scp` (lệnh này sẽ truyền tệp qua bằng SSH) hoặc bằng cách *sao chép* nội dung của khóa công khai và dẫn ống nó vào `ssh` như sau:

```
carol@debian:~/ .ssh$ cat id_ecdsa.pub |ssh ina@192.168.1.77 'cat >> .ssh/authorized_keys'
Password:
```

Khi khóa công khai đã được thêm vào tệp `authorized_keys` trên máy chủ từ xa, ta có thể gặp phải hai tình huống sau khi cố gắng thiết lập một kết nối mới:

- Nếu không cung cấp cụm mật khẩu khi tạo cặp khóa, chúng ta sẽ tự động đăng nhập. Mặc dù thuận tiện nhưng phương pháp này có thể sẽ không an toàn tùy theo từng tình huống:

```
carol@debian:~$ ssh ina@192.168.1.77
Last login: Thu Jun 25 20:31:03 2020 from 192.168.1.55
Have a lot of fun...
ina@halof:~>
```

- Nếu đã cung cấp cụm mật khẩu khi tạo cặp khóa, ta sẽ phải nhập cụm mật khẩu đó trên mọi kết

nối giống như khi nhập mật khẩu. Ngoài khóa công khai, phương pháp này còn bổ sung thêm một lớp bảo mật dưới dạng cụm mật khẩu và từ đó có thể được coi là an toàn hơn. Tuy nhiên, xét về mặt tiện lợi thì nó hoàn toàn giống như việc bạn phải nhập một mật khẩu mỗi khi thiết lập một kết nối. Nếu không sử dụng cụm mật khẩu và ai đó đã tìm được cách để lấy được tệp khóa SSH riêng tư, họ sẽ có quyền truy cập vào mọi máy chủ nơi khóa công khai của chúng ta được cài đặt.

```
carol@debian:~/.ssh$ ssh ina@192.168.1.77
Enter passphrase for key '/home/carol/.ssh/id_ecdsa':
Last login: Thu Jun 25 20:39:30 2020 from 192.168.1.55
Have a lot of fun...
ina@halof:~>
```

Tuy nhiên, có một cách kết hợp giữa bảo mật và tính tiện lợi: sử dụng *tác nhân xác thực SSH* (`ssh-agent`). Tác nhân xác thực cần tạo ra một vỏ riêng của nó và giữ các khóa riêng tư của người dùng — để xác thực khóa công khai — trong bộ nhớ dành cho phần còn lại của phiên. Chúng ta hãy xem cách nó hoạt động một cách chi tiết hơn:

1. Sử dụng `ssh-agent` để khởi động một vỏ Bash mới:

```
carol@debian:~/.ssh$ ssh-agent /bin/bash
carol@debian:~/.ssh$
```

2. Sử dụng lệnh `ssh-add` để thêm khóa riêng tư vào một vùng bộ nhớ an toàn. Nếu đã cung cấp cụm mật khẩu khi tạo cặp khóa — điều này được khuyến nghị để tăng cường bảo mật — chúng ta sẽ được yêu cầu nhập cụm mật khẩu đó:

```
carol@debian:~/.ssh$ ssh-add
Enter passphrase for /home/carol/.ssh/id_ecdsa:
Identity added: /home/carol/.ssh/id_ecdsa (carol@debian)
```

Sau khi danh tính người dùng đã được thêm, chúng ta có thể đăng nhập vào bất kỳ một máy chủ từ xa nào có khóa công khai của chúng ta mà không cần phải nhập lại cụm mật khẩu. Thông thường, trên các máy tính hiện đại, việc thực hiện lệnh này khi khởi động máy tính là khá thông dụng vì nó sẽ vẫn còn trong bộ nhớ cho đến khi máy tính bị tắt (hoặc khóa không được tải xuống theo cách thủ công).

Chúng ta hãy kết thúc phần này bằng cách liệt kê bốn loại thuật toán khóa công khai có thể được chỉ định bằng `ssh-keygen`:

RSA

Được đặt theo tên của những người sáng tạo ra nó là Ron Rivest, Adi Shamir và Leonard Adleman và được xuất bản vào năm 1977. Nó được coi là an toàn và vẫn được sử dụng rộng rãi cho đến ngày nay. Kích thước khóa tối thiểu của nó là 1024 bit (mặc định là 2048).

DSA

Thuật toán chữ ký số đã được chứng minh là không an toàn và không còn được dùng kể từ OpenSSH 7.0. Khóa DSA phải có độ dài chính xác là 1024 bit.

ecdsa

Thuật toán chữ ký số đường cong Elip là một cải tiến trên nền tảng của DSA và vì thế mà được coi là an toàn hơn. Nó sử dụng mật mã đường cong elip. Độ dài khóa ECDSA được xác định bởi một trong ba kích thước đường cong elip có thể tính bằng bit là 256, 384 hoặc 521.

ed25519

Đây là một triển khai của EdDSA — *Thuật toán chữ ký số đường cong Edwards* — sử dụng đường 25519 mạnh hơn. Nó được coi là an toàn nhất trong tất cả các thuật toán. Tất cả các khóa Ed25519 đều có độ dài cố định là 256 bit.

NOTE

Nếu được gọi mà không có chỉ định `-t`, `ssh-keygen` sẽ tạo một cặp khóa RSA theo mặc định.

Vai trò của Khóa máy chủ của Máy chủ OpenSSH

Thư mục cấu hình chung cho OpenSSH được nằm trong thư mục `/etc`:

```
halof:~ # tree /etc/ssh
/etc/ssh
├── moduli
├── ssh_config
├── ssh_host_dsa_key
├── ssh_host_dsa_key.pub
├── ssh_host_ecdsa_key
├── ssh_host_ecdsa_key.pub
├── ssh_host_ed25519_key
├── ssh_host_ed25519_key.pub
├── ssh_host_rsa_key
├── ssh_host_rsa_key.pub
└── sshd_config

0 directories, 11 files
```

Ngoài `moduli` và các tệp cấu hình cho máy khách (`ssh_config`) và máy chủ (`sshd_config`), ta có thể tìm thấy bốn cặp khóa — một cặp khóa cho mỗi một thuật toán được hỗ trợ — được tạo khi máy chủ *OpenSSH* được cài đặt. Như đã nói ở trên, máy chủ sử dụng *khóa máy chủ* này để nhận dạng chính nó với máy khách theo yêu cầu. Mẫu tên của chúng sẽ như sau:

Khóa riêng tư

tiền tố `ssh_host_` + *thuật toán* + hậu tố `key` (ví dụ: `ssh_host_rsa_key`)

Khóa công khai (hoặc vân tay khóa công khai)

tiền tố `ssh_host_` + *thuật toán* + hậu tố `.pub` (ví dụ: `ssh_host_rsa_key.pub`)

NOTE

Vân tay được tạo bằng cách áp dụng hàm băm mật mã cho khóa công khai. Vì vân tay ngắn hơn các khóa mà chúng đề cập đến nên chúng rất hữu ích trong việc đơn giản hóa một số tác vụ quản lý khóa nhất định.

Quyền trên các tệp chứa khóa riêng tư là `0600` hoặc `-rw-----`: chỉ chủ sở hữu (`root`) mới có thể đọc và ghi được. Mặt khác, tất cả các tệp khóa công khai cũng có thể được đọc bởi các thành viên trong nhóm chủ sở hữu và những người khác (`0644` hoặc `-rw-r--r--`):

```
halof:~ # ls -l /etc/ssh/ssh_host_*
-rw----- 1 root root 1381 Dec 21 20:35 /etc/ssh/ssh_host_dsa_key
-rw-r--r-- 1 root root 605 Dec 21 20:35 /etc/ssh/ssh_host_dsa_key.pub
-rw----- 1 root root 505 Dec 21 20:35 /etc/ssh/ssh_host_ecdsa_key
-rw-r--r-- 1 root root 177 Dec 21 20:35 /etc/ssh/ssh_host_ecdsa_key.pub
-rw----- 1 root root 411 Dec 21 20:35 /etc/ssh/ssh_host_ed25519_key
-rw-r--r-- 1 root root 97 Dec 21 20:35 /etc/ssh/ssh_host_ed25519_key.pub
-rw----- 1 root root 1823 Dec 21 20:35 /etc/ssh/ssh_host_rsa_key
-rw-r--r-- 1 root root 397 Dec 21 20:35 /etc/ssh/ssh_host_rsa_key.pub
```

Chúng ta có thể xem vân tay của các khóa bằng cách truyền khóa chuyển `-l` cho `ssh-keygen`. Ta cũng phải cung cấp `-f` để chỉ định đường dẫn tệp chính:

```
halof:~ # ssh-keygen -l -f /etc/ssh/ssh_host_ed25519_key
256 SHA256:8cnPrinC49ZHc+/9Ai5pV+1JfZ4WBRZhd3rD0sc2z1A root@halof (ED25519)
halof:~ # ssh-keygen -l -f /etc/ssh/ssh_host_ed25519_key.pub
256 SHA256:8cnPrinC49ZHc+/9Ai5pV+1JfZ4WBRZhd3rD0sc2z1A root@halof (ED25519)
```

Để xem vân tay cũng như hình ảnh ngẫu nhiên của nó, ta chỉ cần thêm khóa chuyển `-v` như sau:

```
halof:~ # ssh-keygen -lv -f /etc/ssh/ssh_host_ed25519_key.pub
```

```

256 SHA256:8cnPrinC49ZHc+/9Ai5pV+1JfZ4WBRZhd3rD0sc2z1A root@halof (ED25519)
+--[ED25519 256]--+
|           +oo|
|           .+o.|
|           .  ..E.|
|           + .  +.o|
|           S +  + *o|
|           ooo 0o=|
|           . . . =o+.=|
|           = o =oo o=o|
|           o.o +o+ . .o.+|
+-----[SHA256]-----+

```

Đường hầm Cổng SSH

OpenSSH có tính năng chuyển tiếp rất mạnh mẽ, nhờ đó mà lưu lượng truy cập trên cổng nguồn có thể được điều chỉnh — và mã hóa — thông qua tiến trình SSH và sau đó chuyển hướng nó đến một cổng trên máy chủ đích. Cơ chế này được gọi là *tạo đường hầm cho cổng (port tunneling)* hoặc *chuyển tiếp cổng (port forwarding)* và có những ưu điểm quan trọng như sau:

- Nó cho phép ta vượt qua tường lửa để truy cập các cổng trên máy chủ từ xa.
- Nó cho phép truy cập từ bên ngoài vào máy chủ trên mạng riêng của người dùng.
- Nó cung cấp mã hóa cho tất cả các trao đổi dữ liệu.

Nói một cách đại khái, chúng ta có thể phân biệt giữa việc tạo hầm cho cổng cục bộ và từ xa.

Đường hầm Cổng cục bộ

Chúng ta có thể xác định một cổng cục bộ để chuyển tiếp lưu lượng truy cập đến máy chủ đích thông qua tiến trình SSH nằm ở giữa. Tiến trình SSH có thể chạy trên máy chủ cục bộ hoặc một máy chủ từ xa. Ví dụ: nếu vì lý do nào đó mà chúng ta cần tạo đường hầm cho một kết nối tới `www.gnu.org` thông qua SSH bằng cổng 8585 trên máy cục bộ, chúng ta có thể thực hiện như sau:

```

carol@debian:~$ ssh -L 8585:www.gnu.org:80 debian
carol@debian's password:
Linux debian 4.19.0-9-amd64 #1 SMP Debian 4.19.118-2 (2020-04-29) x86_64

The programs included with the Debian GNU/Linux system are free software;
(...)
Last login: Sun Jun 28 13:47:27 2020 from 127.0.0.1

```

Có thể giải thích như sau: với khóa chuyển `-L`, chúng ta đã chỉ định cổng cục bộ `8585` để kết nối với cổng `80 http` trên `www.gnu.org` bằng tiến trình SSH chạy trên `debian` — *máy chủ cục bộ* của chúng ta. Chúng ta có thể viết `ssh -L 8585:www.gnu.org:80 localhost` và đạt được một kết quả tương tự. Bây giờ, nếu sử dụng trình duyệt web để truy cập `http://localhost:8585`, chúng ta sẽ được chuyển tiếp đến `www.gnu.org`. Với mục đích minh họa, chúng ta sẽ sử dụng `lynx` (trình duyệt web ở chế độ văn bản cổ điển):

```
carol@debian:~$ lynx http://localhost:8585
(...)
* Back to Savannah Homepage
  * Not Logged in
  * Login
  * New User
  * This Page
  * Language
  * Clean Reload
  * Printer Version
  * Search
  * _
(...)
```

Nếu cần thực hiện điều tương tự nhưng kết nối thông qua tiến trình SSH chạy trên `halof`, chúng ta sẽ tiến hành như sau:

```
carol@debian:~$ ssh -L 8585:www.gnu.org:80 -Nf ina@192.168.1.77
Enter passphrase for key '/home/carol/.ssh/id_ecdsa':
carol@debian:~$
carol@debian:~$ lynx http://localhost:8585
(...)
* Back to Savannah Homepage
  * Not Logged in
  * Login
  * New User
  * This Page
  * Language
  * Clean Reload
  * Printer Version
  * Search
  * _
(...)
```

Chúng ta cần phải lưu ý ba chi tiết quan trọng trong lệnh:

- Nhờ tùy chọn `-N`, chúng ta đã không đăng nhập vào `halof` mà thay vào đó đã chuyển tiếp cổng.
- Tùy chọn `-f` đã cho SSH biết phải chạy ngầm.
- Chúng ta đã chỉ định người dùng `ina` thực hiện chuyển tiếp: `ina@192.168.1.77`

Đường hầm Cổng từ xa

Trong đường hầm cổng từ xa (hoặc chuyển tiếp cổng ngược), lưu lượng truy cập đến trên một cổng trên máy chủ từ xa sẽ được chuyển tiếp đến tiến trình SSH chạy trên máy chủ cục bộ và từ đó đến cổng được chỉ định trên máy chủ đích (cũng có thể chính là máy cục bộ của chúng ta). Ví dụ: giả sử chúng ta cần cho phép ai đó từ bên ngoài mạng truy cập vào máy chủ web Apache chạy trên máy chủ cục bộ thông qua cổng `8585` của máy chủ SSH chạy trên `halof` (`192.168.1.77`), chúng ta có thể tiến hành như sau:

```
carol@debian:~$ ssh -R 8585:localhost:80 -Nf ina@192.168.1.77
Enter passphrase for key '/home/carol/.ssh/id_ecdsa':
carol@debian:~$
```

Giờ đây, bất kỳ ai thiết lập kết nối tới `halof` trên cổng `8585` cũng sẽ thấy trang chủ mặc định Apache2 của Debian:

```
carol@debian:~$ lynx 192.168.1.77:8585
(...)
Apache2 Debian Default
Page: It works (p1 of 3)
Debian Logo Apache2 Debian Default Page
It works!
This is the default welcome page used to test the correct operation of the Apache2 server
after
installation on Debian systems. If you can read this page, it means that the Apache HTTP
server
installed at this site is working properly. You should replace this file (located at
/var/www/html/index.html) before continuing to operate your HTTP server.
(...)
```

NOTE

Có một loại chuyển tiếp cổng thứ ba phức tạp hơn nằm ngoài phạm vi của bài học này là *chuyển tiếp cổng động* (*dynamic port forwarding*). Thay vì tương tác với một cổng duy nhất, kiểu chuyển tiếp này sử dụng nhiều giao tiếp TCP khác nhau trên

nhiều cổng.

Đường hầm X11

Bây giờ chúng ta đã hiểu về đường hầm cổng, chúng ta có thể kết thúc bài học này bằng cách thảo luận về đường hầm X11 (còn được gọi là *X11forwarding*). Thông qua đường hầm X11, *Hệ thống X Window* trên máy chủ từ xa sẽ được chuyển tiếp đến máy cục bộ của chúng ta. Để làm được điều này, ta chỉ cần truyền cho `ssh` tùy chọn `-X`:

```
carol@debian:~$ ssh -X ina@halof
...
```

Bây giờ, chúng ta có thể khởi chạy một ứng dụng đồ họa như trình duyệt web `firefox` với kết quả sau: ứng dụng sẽ chạy trên máy chủ từ xa nhưng màn hình của nó sẽ được chuyển tiếp đến máy chủ cục bộ của chúng ta.

Thay vào đó, nếu ta bắt đầu một phiên SSH mới với tùy chọn `-x`, *X11forwarding* sẽ bị vô hiệu hóa. Nếu thử khởi động `firefox` ngay bây giờ, chúng ta sẽ gặp lỗi như sau:

```
carol@debian:~$ ssh -x ina@halof
carol@192.168.0.106's password:
(...)
ina@halof:~$ firefox

(firefox-esr:1779): Gtk-WARNING **: 18:45:45.603: Locale not supported by C library.
  Using the fallback 'C' locale.
Error: no DISPLAY environment variable specified
```

NOTE

Ba chỉ thị cấu hình liên quan đến chuyển tiếp cổng cục bộ, chuyển tiếp cổng từ xa và chuyển tiếp X11 lần lượt là `AllowTcpForwarding`, `GatewayPorts` và `X11Forwarding`. Để biết thêm thông tin, hãy nhập man `ssh_config` và/hoặc man `sshd_config`.

Bài tập Hướng dẫn

1. Hãy đăng nhập với tư cách là người dùng `sonya` trên máy khách của bạn và thực hiện các tác vụ SSH sau trên máy chủ từ xa `halof`:

- Thực thi lệnh để liệt kê nội dung của `~/ .ssh` với tư cách là người dùng `serena` trên máy chủ từ xa; sau đó quay trở lại cửa sổ dòng lệnh cục bộ của bạn.

- Đăng nhập với tư cách là người dùng `serena` trên máy chủ từ xa.

- Đăng nhập với tư cách là người dùng `sonya` trên máy chủ từ xa.

- Xóa tất cả các khóa thuộc về `halof` khỏi tệp `~/ .ssh/known_hosts` cục bộ của bạn.

- Trên máy khách của bạn, tạo cặp khóa `ecdsa` gồm 256 bit.

- Trên máy khách của bạn, tạo cặp khóa `ed25519` gồm 256 bit.

2. Hãy thực hiện các bước sau theo đúng thứ tự để thiết lập kết nối SSH bằng *tác nhân xác thực SSH*:

- Trên máy khách, khởi động một vỏ Bash mới cho *tác nhân xác thực* bằng `ssh-agent /bin/bash`.
- Trên máy khách, tạo một cặp khóa bằng `ssh-keygen`.
- Trên máy khách, thêm khóa riêng tư của bạn vào một vùng bộ nhớ an toàn bằng `ssh-add`.
- Thêm khóa công khai của máy khách vào tệp `~/ .ssh/authorized_keys` của người dùng mà bạn muốn sử dụng để đăng nhập trên máy chủ từ xa.
- Nếu chưa tồn tại, hãy tạo `~/ .ssh` cho người dùng bạn muốn sử dụng để đăng nhập trên máy chủ.
- Kết nối với máy chủ từ xa.

Thứ tự đúng sẽ là:

Bước 1:	
Bước 2:	
Bước 3:	
Bước 4:	
Bước 5:	
Bước 6:	

3. Về *chuyển tiếp công*, tùy chọn và chỉ thị nào được sử dụng cho các loại đường hầm sau:

Loại đường hầm	Tùy chọn	Chỉ thị
Cục bộ		
Từ xa hoặc đảo ngược		
X		

4. Giả sử bạn gõ lệnh `ssh -L 8888:localhost:80 -Nf ina@halof` vào cửa sổ dòng lệnh của máy khách. Vẫn trên máy khách, bạn trở một trình duyệt tới `http://localhost:8888`. Bạn sẽ nhận được kết quả gì?

Bài tập Mở rộng

1. Liên quan đến chỉ thị bảo mật SSH:

- Lệnh nào được sử dụng trong `/etc/ssh/sshd_config` để kích hoạt phiên đăng nhập cho `root`:

- Bạn sẽ sử dụng lệnh nào trong `/etc/ssh/sshd_config` để chỉ định một tài khoản cục bộ chấp nhận các kết nối SSH:

2. Khi sử dụng cùng một người dùng trên cả máy khách và máy chủ, bạn có thể sử dụng lệnh `ssh` nào để chuyển khóa công khai của máy khách sang máy chủ để bạn có thể đăng nhập thông qua xác thực khóa công khai?

3. Hãy tạo hai đường hầm cổng cục bộ bằng một lệnh duy nhất để chuyển tiếp các cổng cục bộ không có đặc quyền 8080 và 8585 thông qua máy chủ từ xa `halof` tương ứng tới các trang web `www.gnu.org` và `www.melpa.org`. Hãy sử dụng người dùng `ina` trên máy chủ từ xa và đừng quên sử dụng các khóa chuyển `-Nf`:

Tóm tắt

Trong bài học này, chúng ta đã thảo luận về *OpenSSH 2* sử dụng giao thức *Secure Shell* để mã hóa thông tin liên lạc giữa máy chủ và máy khách. Chúng ta đã học được:

- cách đăng nhập vào máy chủ từ xa.
- cách thực hiện lệnh từ xa.
- cách tạo cặp khóa.
- cách thiết lập các phiên đăng nhập dựa trên khóa.
- cách sử dụng *tác nhân xác thực* để tăng cường bảo mật và tính tiện dụng.
- các thuật toán khóa công khai được hỗ trợ bởi *OpenSSH*: `RSA`, `DSA`, `ecdsa`, `ed25519`.
- vai trò của khóa máy chủ *OpenSSH*.
- cách tạo đường hầm cổng: cục bộ, từ xa và X.

Các lệnh sau đã được thảo luận trong bài học này:

ssh

Đăng nhập hoặc thực thi lệnh trên máy từ xa.

ssh-keygen

Tạo, quản lý và chuyển đổi khóa xác thực.

ssh-agent

Tác nhân xác thực OpenSSH.

ssh-add

Thêm danh tính của khóa riêng tư vào tác nhân xác thực.

Đáp án Bài tập Hướng dẫn

1. Hãy đăng nhập với tư cách là người dùng `sonya` trên máy khách của bạn và thực hiện các tác vụ SSH sau trên máy chủ từ xa `halof`:

- Thực thi lệnh để liệt kê nội dung của `~/ .ssh` với tư cách là người dùng `serena` trên máy chủ từ xa; sau đó quay trở lại cửa sổ dòng lệnh cục bộ của bạn.

```
ssh serena@halof ls .ssh
```

- Đăng nhập với tư cách là người dùng `serena` trên máy chủ từ xa.

```
ssh serena@halof
```

- Đăng nhập với tư cách là người dùng `sonya` trên máy chủ từ xa.

```
ssh halof
```

- Xóa tất cả các khóa thuộc về `halof` khỏi tệp `~/ .ssh/known_hosts` cục bộ của bạn.

```
ssh-keygen -R halof
```

- Trên máy khách của bạn, tạo cặp khóa `ecdsa` gồm 256 bit.

```
ssh-keygen -t ecdsa -b 256
```

- Trên máy khách của bạn, tạo cặp khóa `ed25519` gồm 256 bit.

```
ssh-keygen -t ed25519
```

2. Hãy thực hiện các bước sau theo đúng thứ tự để thiết lập kết nối SSH bằng *tác nhân xác thực SSH*:

- Trên máy khách, khởi động một vỏ Bash mới cho *tác nhân xác thực* bằng `ssh-agent /bin/bash`.
- Trên máy khách, tạo một cặp khóa bằng `ssh-keygen`.
- Trên máy khách, thêm khóa riêng tư của bạn vào một vùng bộ nhớ an toàn bằng `ssh-add`.

- Thêm khóa công khai của máy khách vào tệp `~/.ssh/authorized_keys` của người dùng mà bạn muốn sử dụng để đăng nhập trên máy chủ từ xa.
- Nếu chưa tồn tại, hãy tạo `~/.ssh` cho người dùng bạn muốn sử dụng để đăng nhập trên máy chủ.
- Kết nối với máy chủ từ xa.

Thứ tự đúng sẽ là:

Bước 1:	Trên máy khách, tạo cặp khóa bằng <code>ssh-keygen</code> .
Bước 2:	* Nếu chưa tồn tại, hãy tạo <code>~/.ssh</code> cho người dùng bạn muốn sử dụng để đăng nhập trên máy chủ.
Bước 3:	* Thêm khóa công khai của máy khách vào tệp <code>~/.ssh/authorized_keys</code> của người dùng mà bạn muốn sử dụng để đăng nhập trên máy chủ từ xa.
Bước 4:	* Trên máy khách, khởi động một vỏ Bash mới cho <i>tác nhân xác thực</i> bằng <code>ssh-agent /bin/bash</code> .
Bước 5:	* Trên máy khách, thêm khóa riêng tư của bạn vào một vùng bộ nhớ an toàn bằng <code>ssh-add</code> .
Bước 6:	* Kết nối với máy chủ từ xa.

3. Về *chuyển tiếp cổng*, tùy chọn và lệnh nào được sử dụng cho các loại đường hầm sau:

Loại đường hầm	Tùy chọn	Chỉ thị
Cục bộ	-L	AllowTcpForwarding
Từ xa hoặc đảo ngược	-R	GatewayPorts
X	-X	X11Forwarding

4. Giả sử bạn gõ lệnh `ssh -L 8888:localhost:80 -Nf ina@halof` vào cửa sổ dòng lệnh của máy khách. Vẫn trên máy khách, bạn trở một trình duyệt tới `http://localhost:8888`. Bạn sẽ nhận được kết quả gì?

Trang chủ của máy chủ web `halof` vì `localhost` được hiểu dưới góc độ của máy chủ.

Đáp án Bài tập Mở rộng

1. Liên quan đến chỉ thị bảo mật SSH:

- Lệnh nào được sử dụng trong `/etc/ssh/sshd_config` để kích hoạt phiên đăng nhập cho `root`:

```
PermitRootLogin
```

- Bạn sẽ sử dụng lệnh nào trong `/etc/ssh/sshd_config` để chỉ định một tài khoản cục bộ chấp nhận các kết nối SSH:

```
AllowUsers
```

2. Khi sử dụng cùng một người dùng trên cả máy khách và máy chủ, bạn có thể sử dụng lệnh `ssh` nào để chuyển khóa công khai của máy khách sang máy chủ để bạn có thể đăng nhập thông qua xác thực khóa công khai?

```
ssh-copy-id
```

3. Hãy tạo hai đường hầm cổng cục bộ bằng một lệnh duy nhất để chuyển tiếp các cổng cục bộ không có đặc quyền 8080 và 8585 thông qua máy chủ từ xa `halof` tương ứng tới các trang web `www.gnu.org` và `www.melpa.org`. Hãy sử dụng người dùng `ina` trên máy chủ từ xa và đừng quên sử dụng các khóa chuyển `-Nf`:

```
ssh -L 8080:www.gnu.org:80 -L 8585:www.melpa.org:80 -Nf ina@halof
```




110.3 Bài 2

Chứng chỉ:	LPIC-1
Phiên bản:	5.0
Chủ đề:	110 Bảo mật
Mục tiêu:	110.3 Bảo mật Dữ liệu bằng Mã hóa
Bài:	2 trên 2

Giới thiệu

Ở bài học trước, chúng ta đã học cách sử dụng *OpenSSH* để mã hóa các phiên đăng nhập từ xa cũng như mọi hoạt động trao đổi thông tin diễn ra tiếp sau đó. Có thể có một số trường hợp mà chúng ta sẽ phải mã hóa tệp hoặc email để chúng đến tay người nhận một cách an toàn và tránh khỏi những con mắt tò mò. Chúng ta cũng có thể cần phải ký chữ ký điện tử vào các tệp hoặc tin nhắn đó để ngăn việc chúng bị giả mạo.

Một công cụ tuyệt vời dành cho những trường hợp sử dụng này là *GNU Privacy Guard* (hay còn gọi là *GnuPG*, hoặc đơn giản là *GPG*). Đây là một triển khai mã nguồn mở và miễn phí của riêng *Pretty Good Privacy (PGP)*. *GPG* sử dụng tiêu chuẩn *OpenPGP* như được xác định trong RFC 4880 bởi *Nhóm công tác OpenPGP* của *Lực lượng Kỹ thuật Internet đặc nhiệm (IETF)*. Trong bài học này, chúng ta sẽ xem xét những kiến thức cơ bản về *GNU Privacy Guard*.

Thực hiện cấu hình, sử dụng và thu hồi GnuPG cơ bản

Cũng giống như *SSH*, cơ chế cơ bản của *GPG* là *mật mã bất đối xứng* hoặc *mật mã khóa công khai*. Người dùng sẽ tạo một cặp khóa được tạo thành từ một *khóa riêng tư* và một *khóa công khai*. Các khóa này có liên quan về mặt toán học sao cho những gì được mã hóa bởi khóa này chỉ có thể

được giải mã bởi khoá kia. Để giao tiếp diễn ra thành công, người dùng phải gửi khoá công khai của họ cho người nhận được xác định.

Cấu hình và cách sử dụng GnuPG

Lệnh để làm việc với GPG là `gpg`. Chúng ta có thể truyền cho nó một số tùy chọn để thực hiện các tác vụ khác nhau. Hãy cùng bắt đầu bằng cách tạo một cặp khóa với tư cách là người dùng `carol`. Để làm được điều này, chúng ta sẽ sử dụng lệnh `gpg --gen-key`:

```
carol@debian:~$ gpg --gen-key
gpg (GnuPG) 2.2.12; Copyright (C) 2018 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: directory '/home/carol/.gnupg' created
gpg: keybox '/home/carol/.gnupg/pubring.kbx' created
Note: Use "gpg --full-generate-key" for a full featured key generation dialog.

GnuPG needs to construct a user ID to identify your key.

Real name:

(...)
```

Sau khi thông báo cho người dùng — bên cạnh một số hoạt động khác — rằng thư mục cấu hình `~/.gnupg` và khóa công khai `~/.gnupg/pubring.kbx` đã được tạo, `gpg` sẽ tiếp tục yêu cầu người dùng cung cấp tên thực và địa chỉ email:

```
(...)
Real name: carol
Email address: carol@debian
You selected this USER-ID:
    "carol <carol@debian>"

Change (N)ame, (E)mail, or (O)kay/(Q)uit?
```

Nếu đồng ý với kết quả `USER-ID` và nhấn `o`, chúng ta sẽ được yêu cầu nhập một cụm mật khẩu (khuyến nghị rằng cụm mật khẩu đó phải ở một mức độ phức tạp nhất định):

```
| Please enter the passphrase to |
```

```
| protect your new key
|
| Passphrase: |
(...)

```

Một số thông báo cuối cùng sẽ được hiển thị cho người dùng về việc tạo các tệp khác cũng như chính các khóa, sau đó chúng ta sẽ hoàn tất quá trình tạo khóa:

```
(...)
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: /home/carol/.gnupg/trustdb.gpg: trustdb created
gpg: key 19BBEFD16813034E marked as ultimately trusted
gpg: directory '/home/carol/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/carol/.gnupg/openpgp-
revocs.d/D18FA0021F644CDAF57FD0F919BBEFD16813034E.rev'
public and secret key created and signed.

pub  rsa3072 2020-07-03 [SC] [expires: 2022-07-03]
     D18FA0021F644CDAF57FD0F919BBEFD16813034E
uid  carol <carol@debian>
sub  rsa3072 2020-07-03 [E] [expires: 2022-07-03]

```

Bây giờ, chúng ta có thể xem bên trong thư mục `~/ .gnupg` (thư mục cấu hình của GPG) có gì:

```
carol@debian:~/ .gnupg$ ls -l
total 16
drwx----- 2 carol carol 4096 Jul  3 23:34 openpgp-revocs.d
drwx----- 2 carol carol 4096 Jul  3 23:34 private-keys-v1.d
-rw-r--r--  1 carol carol 1962 Jul  3 23:34 pubring.kbx
-rw-----  1 carol carol 1240 Jul  3 23:34 trustdb.gpg

```

Hãy cùng giải thích về mục đích của từng tệp:

openpgp-revocs.d

Chứng chỉ thu hồi đã được tạo cùng với cặp khóa sẽ được lưu giữ ở đây. Các quyền trên thư mục này khá là hạn chế vì bất kỳ ai có quyền truy cập vào chứng chỉ đều có thể thu hồi khóa (thông tin thêm về việc thu hồi khóa sẽ có trong tiểu mục tiếp theo).

private-keys-v1.d

Đây là thư mục lưu giữ các khóa riêng tư của người dùng, vì thế mà các quyền trên nó rất hạn chế.

pubring.kbx

Đây là khóa công khai của người dùng. Nó lưu trữ khóa công khai của riêng người dùng cũng như bất kỳ một khóa công khai nào khác được nhập.

trustdb.gpg

Cơ sở dữ liệu tin cậy. Điều này liên quan đến khái niệm *Mạng lưới tin cậy* (nằm ngoài phạm vi của bài học này).

NOTE

Sự xuất hiện của *GnuPG 2.1* đã mang đến một số thay đổi đáng kể như sự biến mất của các tệp `secring.gpg` và `pubring.gpg` và tương ứng thay vào đó là các tệp `private-keys-v1.d` và `pubring.kbx`.

Khi cặp khóa đã được tạo, chúng ta có thể xem khóa công khai của mình bằng `gpg --list-keys` — lệnh này sẽ hiển thị nội dung khóa công khai của chúng ta:

```
carol@debian:~/gnupg$ gpg --list-keys
/home/carol/.gnupg/pubring.kbx
-----
pub   rsa3072 2020-07-03 [SC] [expires: 2022-07-03]
      D18FA0021F644CDAF57FD0F919BBEFD16813034E
uid           [ultimate] carol <carol@debian>
sub   rsa3072 2020-07-03 [E] [expires: 2022-07-03]
```

Chuỗi thập lục phân `D18FA0021F644CDAF57FD0F919BBEFD16813034E` chính là *vân tay khóa công khai* của chúng ta.

NOTE

Ngoài `USER-ID` (trong ví dụ là `carol`) thì chúng ta còn có `KEY-ID`. `KEY-ID` bao gồm 8 chữ số thập lục phân cuối cùng trong vân tay khóa công khai của bạn (`6813034E`). Bạn có thể kiểm tra vân tay khóa của mình bằng lệnh `gpg --fingerprint USER-ID`.

Phân phối và thu hồi Khóa

Khi đã có khóa công khai, ta nên lưu (hay là *xuất*) nó vào một tệp để làm cho khoá trở nên khả dụng đối với những người nhận trong tương lai. Sau đó, họ sẽ có thể sử dụng tệp này để mã hóa các tệp hoặc tin nhắn dành cho chúng ta (vì chúng ta là người duy nhất sở hữu khóa riêng tư nên cũng sẽ là người duy nhất có thể giải mã và đọc chúng). Tương tự như vậy, người nhận của chúng

ta cũng sẽ sử dụng nó để giải mã và xác minh các tin nhắn/tệp được mã hóa hoặc ký tên của chúng ta. Lệnh được sử dụng là `gpg --export`, theo sau là `USER-ID` và chuyển hướng đến tên tệp đầu ra được chọn:

```
carol@debian:~/.gnupg$ gpg --export carol > carol.pub.key
carol@debian:~/.gnupg$ ls
carol.pub.key  openpgp-revocs.d  private-keys-v1.d  pubring.kbx  trustdb.gpg
```

NOTE

Việc truyền tùy chọn `-a` hoặc `--armor` cho `gpg --export` (ví dụ như `gpg --export --armor carol > carol.pub.key`) sẽ kết xuất đầu ra dưới dạng văn bản mã ASCII (thay vì định dạng OpenPGP nhị phân mặc định) có thể được gửi qua email một cách an toàn.

Như đã lưu ý ở trên, bây giờ chúng ta sẽ phải gửi tệp khóa công khai của mình (`carol.pub.key`) tới người nhận mà ta muốn trao đổi thông tin. Ví dụ: hãy gửi tệp khóa công khai tới `ina` trên máy chủ từ xa `halof` bằng cách sử dụng `scp` (*secure copy*):

```
carol@debian:~/.gnupg$ scp carol.pub.key ina@halof:/home/ina/
Enter passphrase for key '/home/carol/.ssh/id_ecdsa':
carol.pub.key
100% 1740  775.8KB/s  00:00
carol@debian:~/.gnupg$
```

`ina` hiện đã có được quyền sở hữu `carol.pub.key`. Cô ấy sẽ sử dụng nó để mã hóa một tệp và gửi nó tới `carol` trong phần tiếp theo.

NOTE

Một cách phân phối khóa công khai khác là thông qua việc sử dụng *máy chủ khóa* (*key servers*): chúng ta sẽ tải khóa công khai của mình lên máy chủ bằng lệnh `gpg --keyserver keyserver-name --send-keys KEY-ID` và những người dùng khác sẽ có thể nhận (hay còn gọi là *nhập*) chúng với lệnh `gpg --keyserver keyserver-name --recv-keys KEY-ID`.

Hãy cùng kết thúc phần này bằng cách thảo luận về việc thu hồi khóa. Việc thu hồi khóa nên được sử dụng khi khóa riêng tư của chúng ta bị xâm phạm hoặc hết hạn. Bước đầu tiên là tạo chứng chỉ thu hồi bằng cách truyền cho `gpg` tùy chọn `--gen-revoke`, theo sau là `USER-ID`. Ta có thể đặt trước `--gen-revoke` tùy chọn `--output`, theo sau là đặc tả tên tệp đích để lưu chứng chỉ kết quả vào một tệp (thay vì in nó trên màn hình cửa sổ dòng lệnh). Các thông báo đầu ra trong suốt quá trình thu hồi sẽ khá dễ hiểu:

```
sonya@debian:~/.gnupg$ gpg --output revocation_file.asc --gen-revoke sonya
```

```
sec rsa3072/0989EB7E7F9F2066 2020-07-03 sonya <sonya@debian>
```

```
Create a revocation certificate for this key? (y/N) y
```

```
Please select the reason for the revocation:
```

- 0 = No reason specified
- 1 = Key has been compromised
- 2 = Key is superseded
- 3 = Key is no longer used
- Q = Cancel

```
(Probably you want to select 1 here)
```

```
Your decision? 1
```

```
Enter an optional description; end it with an empty line:
```

```
> My laptop was stolen.
```

```
>
```

```
Reason for revocation: Key has been compromised
```

```
My laptop was stolen.
```

```
Is this okay? (y/N) y
```

```
ASCII armored output forced.
```

```
Revocation certificate created.
```

Please move it to a medium which you can hide away; if Mallory gets access to this certificate he can use it to make your key unusable. It is smart to print this certificate and store it away, just in case your media become unreadable. But have some caution: The print system of your machine might store the data and make it available to others!

Chúng chỉ thu hồi đã được lưu vào tệp `revocation_file.asc` (asc cho định dạng ASCII):

```
sonya@debian:~/.gnupg$ ls
openpgp-revocs.d private-keys-v1.d pubring.kbx revocation_file.asc trustdb.gpg
sonya@debian:~/.gnupg$ cat revocation_file.asc
-----BEGIN PGP PUBLIC KEY BLOCK-----
Comment: This is a revocation certificate

iQHDBCABCgAtFiEEiIVjfDnnpieFi0wvnlcN6yLCeHEFA18ASx4PHQJzdG9sZW4g
bGFwdG9wAAoJEJ5XDesiwnhxT9YMAKkjQiMpo9Uyiy9hyvukPPSrLcmtAGLk4pKS
pLZfzA5kxa+HPQwBgIAEvfNRR6VMxqXUgUGYC/IAYQQM62oNACy2PCPrxyJNgVF7
8l4mMZKvW++5ikjZwyg6WVW0+w6oro9qruJFjcu752p4T+9gsHV2r+KRqcPQe
aZ65sAvsBJlcsUDZqfWUXg2kQp9mNPCdQuqvDaKRgNCHA1zbzNFzXWVd2X5RgFo5
nY+tUP8ZQA9DTQPBLPcggICmfLopMPZYB2bft5geb2mMi2oNpf9CNPdQkdccimNV
aRjqdUP9C89PwTafBQkQiONlsR/dWTFcqrG5KOWQPA7xjeMV8wretdEgsyTxqHp
v1iRzWjshiJCKBXXvz7wSmQrJ40fiMDHeS4ipR0AYd08QCzm0zmcFQKikGSHGMy1
```

```
z/YRl1ttD6NZIKjf1TD0nTrFnRvPdsZ0LKYSArbfqNrHRBQkgirOD4JPI1tYKTffq
i0eZFx25K+fj2+0AJjvrbe4HDo5m+Q==
=umI8
-----END PGP PUBLIC KEY BLOCK-----
```

Để thu hồi khóa riêng tư một cách hiệu quả, ta cần hợp nhất chứng chỉ với khóa. Việc này có thể được thực hiện bằng cách nhập tệp chứng chỉ thu hồi vào vòng khóa (keyring):

```
sonya@debian:~/.gnupg$ gpg --import revocation_file.asc
gpg: key 9E570DEB22C27871: "sonya <sonya@debian>" revocation certificate imported
gpg: Total number processed: 1
gpg:    new key revocations: 1
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid: 1  signed: 0  trust: 0-, 0q, 0n, 0m, 0f, 1u
gpg: next trustdb check due at 2022-07-04
```

Nếu liệt kê các khóa ngay bây giờ, chúng ta sẽ được thông báo về khóa đã bị thu hồi của mình:

```
sonya@debian:~/.gnupg$ gpg --list-keys
/home/sonya/.gnupg/pubring.kbx
pub  rsa3072 2020-07-04 [SC] [revoked: 2020-07-04]
     8885637C39E7A627858B4C2F9E570DEB22C27871
uid           [ revoked] sonya <sonya@debian>
```

Cuối cùng — nhưng không kém phần quan trọng — hãy đảm bảo tính khả dụng cho khoá được thu hồi đối với tất cả các bên có khoá công khai liên kết với nó (bao gồm cả máy chủ khóa).

Sử dụng GPG để mã hóa, giải mã, ký và xác minh tệp

Trong phần trước, `carol` đã gửi khóa công khai của mình cho `ina`. Bây giờ, chúng ta sẽ sử dụng nó để thảo luận về cách GPG có thể mã hóa, giải mã, ký và xác minh tệp.

Mã hóa và giải mã Tệp

Đầu tiên, `ina` sẽ phải nhập khóa công khai của `carol` (`carol.pub.key`) vào vòng khóa của mình để có thể bắt đầu làm việc với nó:

```
ina@halof:~> gpg --import carol.pub.key
gpg: /home/ina/.gnupg/trustdb.gpg: trustdb created
gpg: key 19BBEFD16813034E: public key "carol <carol@debian>" imported
```

```

gpg: Total number processed: 1
gpg:             imported: 1
ina@halof:~> gpg --list-keys
/home/ina/.gnupg/pubring.kbx
-----
pub  rsa3072 2020-07-03 [SC] [expires: 2022-07-03]
     D18FA0021F644CDAF57FD0F919BBEFD16813034E
uid      [ unknown] carol <carol@debian>
sub  rsa3072 2020-07-03 [E] [expires: 2022-07-03]

```

Tiếp theo, chúng ta sẽ tạo một tệp bằng cách viết một đoạn văn bản vào đó và sau đó mã hóa nó bằng gpg (vì đã không ký khóa của carol nên chúng ta sẽ được hỏi xem liệu ta có muốn sử dụng khóa đó hay không):

```

ina@halof:~> echo "This is the message ..." > unencrypted-message
ina@halof:~> gpg --output encrypted-message --recipient carol --armor --encrypt unencrypted-
message
gpg: 0227347CC92A5CB1: There is no assurance this key belongs to the named user
sub  rsa3072/0227347CC92A5CB1 2020-07-03 carol <carol@debian>
     Primary key fingerprint: D18F A002 1F64 4CDA F57F D0F9 19BB EFD1 6813 034E
     Subkey fingerprint: 9D89 1BF9 39A4 C130 E44B 1135 0227 347C C92A 5CB1

It is NOT certain that the key belongs to the person named
in the user ID.  If you really know what you are doing,
you may answer the next question with yes.

Use this key anyway? (y/N) y

```

Hãy cùng chia nhỏ lệnh gpg:

--output encrypted-message

Đặc tả tên tệp cho phiên bản được mã hóa của tệp gốc (trong ví dụ là `encrypted-message`).

--recipient carol

Thông số USER-ID của người nhận (trong ví dụ là `carol`). Nếu không được cung cấp, GnuPG sẽ tự yêu cầu (trừ khi người nhận mặc định - `default-recipient` - được chỉ định).

--armor

Tùy chọn này tạo ra đầu ra dưới dạng văn bản mã ASCII có thể được sao chép vào email.

--encrypt unencrypted-message

Đặc tả tên tệp của tệp gốc cần mã hóa.

Bây giờ, chúng ta có thể gửi thông điệp được mã hóa (`encrypted-message`) tới `carol` trên `debian` bằng cách sử dụng `scp`:

```
ina@halof:~> scp encrypted-message carol@debian:/home/carol/
carol@debian's password:
encrypted-message                                100% 736
1.8MB/s 00:00
```

Nếu bây giờ ta đăng nhập bằng `carol` và thử đọc thông báo được mã hóa, chúng ta sẽ xác nhận được rằng nó thực sự đã được mã hóa và — do đó — không thể đọc được:

```
carol@debian:~$ cat encrypted-message
-----BEGIN PGP MESSAGE-----

hQGMAwInNHZJKlyxAQv/brJ8Ubs/xya35sbv6kdRkm1C70NLxL30ueWA4mCs0Y/P
GBna6ZEUCrMEgl/rCyByj3Yq74kuiTmzxAIRUDdvHfj0Ttr0WjVAqIn/fPSfMkjk
dTxKo1i55tLJ+sJ17dGMZDcNBInBTP4U1atuN71A5w7vH+XpcesRcFQLKiS0mYTt
F7SN3/5x5J6io4ISn+b0KbJgiJNNx+Ne/ub4Uzk4NlK7tmBklyC1VRualtxcG7R9
1k1BPYSld6fTdDwT1Y4MofpyILAiGMZvUR1RXauEKf70IzwC5gWU+UQPSgeCdKQu
X7QL0ZIBS0Ug2XKr01k93lmDjf8PwsRIm16n/hNe1a0BA3HMP0b60zv1gFeEsFvC
IxhUYPb+rfuNFTMEB7xIO94AAmWB9N4qknMxdDqNE8WhA728P1w6y8L2ngsp1Y15
MR41IFDpljA/CcVh4BXVe9j0TdFWDUKrFMfaIfcPQwKLXEYJp19XYIaaEazk0s5D
W4pENN0Y0cX0KWyAYX6r018BF0rq/HMenQwqAVXMG3s8ATuU0eqjBbR1x1qCvRQP
CR/3V73aQwc2j5ioQmhWYppqxiro0yKX2Ar/E6rZyJtJYrq+CUk803JoBaudknNFj
pwuRwF1amwnSZ/MZ/9kMKQ==
=g1jw
-----END PGP MESSAGE-----
```

Tuy nhiên, vì đang sở hữu khóa riêng tư nên chúng ta có thể dễ dàng giải mã tin nhắn bằng cách truyền cho `gpg` tùy chọn `--decrypt`, theo sau là đường dẫn đến tệp được mã hóa (cụm mật khẩu của khóa riêng tư sẽ được yêu cầu):

```
carol@debian:~$ gpg --decrypt encrypted-message
gpg: encrypted with 3072-bit RSA key, ID 0227347CC92A5CB1, created 2020-07-03
"carol <carol@debian>"
This is the message ...
```

Chúng ta cũng có thể chỉ định tùy chọn `--output` để lưu tin nhắn vào một tệp mới không được mã hóa:

```
carol@debian:~$ gpg --output unencrypted-message --decrypt encrypted-message
gpg: encrypted with 3072-bit RSA key, ID 0227347CC92A5CB1, created 2020-07-03
"carol <carol@debian>"
carol@debian:~$ cat unencrypted-message
This is the message ...
```

Ký và xác minh Tệp

Ngoài việc mã hóa, GPG còn có thể được sử dụng để ký các tệp. Chúng ta sẽ cần tới tùy chọn `--sign` ở đây. Hãy bắt đầu bằng cách tạo một tin nhắn mới (`message`) và ký nó bằng tùy chọn `--sign` (cụm mật khẩu khóa riêng tư sẽ được yêu cầu):

```
carol@debian:~$ echo "This is the message to sign ..." > message
carol@debian:~$ gpg --output message.sig --sign message
(...)
```

Phân tích lệnh `gpg`:

`--output message`

Đặc tả tên tệp của phiên bản tệp gốc được ký (trong ví dụ của chúng ta là `message.sig`).

`--sign message`

Đường dẫn đến tệp gốc.

NOTE

Bằng cách sử dụng `--sign`, tài liệu sẽ được nén và sau đó được ký. Đầu ra sẽ ở định dạng nhị phân.

Tiếp theo, chúng ta sẽ chuyển tệp sang cho `ina` trên `halof` bằng cách sử dụng `scp message.sig ina@halof:/home/ina`. Trở lại `ina` trên `halof`, bây giờ chúng ta đã có thể xác minh nó bằng cách sử dụng tùy chọn `--verify`:

```
ina@halof:~> gpg --verify message.sig
gpg: Signature made Sat 04 jul 2020 14:34:41 CEST
gpg: using RSA key D18FA0021F644CDAF57FD0F919BBEFD16813034E
gpg: Good signature from "carol <carol@debian>" [unknown]
(...)
```

Nếu cũng muốn đọc tệp, ta sẽ phải giải mã nó thành một tệp mới (trong trường hợp của chúng ta là message) bằng cách sử dụng tùy chọn `--output`:

```
ina@halof:~> gpg --output message --decrypt message.sig
gpg: Signature made Sat 04 Jul 2020 14:34:41 CEST
gpg:          using RSA key D18FA0021F644CDAF57FD0F919BBEFD16813034E
gpg: Good signature from "carol <carol@debian>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: D18F A002 1F64 4CDA F57F D0F9 19BB EFD1 6813 034E
ina@halof:~> cat message
This is the message to sign ...
```

PGP-Agent

Chúng ta sẽ kết thúc bài học này bằng cách đề cập ngắn gọn đến `gpg-agent`. `gpg-agent` là một trình nền quản lý các khóa riêng tư cho GPG (nó được khởi động theo yêu cầu của `gpg`). Để xem tóm tắt các tùy chọn hữu ích nhất, hãy chạy `gpg-agent --help` hoặc `gpg-agent -h`:

```
carol@debian:~$ gpg-agent --help
gpg-agent (GnuPG) 2.2.4
libgcrypt 1.8.1
Copyright (C) 2017 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Syntax: gpg-agent [options] [command [args]]
Secret key management for GnuPG

Options:

  --daemon          run in daemon mode (background)
  --server          run in server mode (foreground)
  --supervised      run in supervised mode
  -v, --verbose     verbose
  -q, --quiet       be somewhat more quiet
  -s, --sh          sh-style command output
  -c, --csh         csh-style command output
  (...)            ...
```

NOTE Để biết thêm thông tin, hãy tham khảo trang hướng dẫn `gpg-agent`.

Bài tập Hướng dẫn

1. Hãy hoàn thành bảng bằng cách cung cấp tên tệp chính xác:

Mô tả	Tên tệp
Cơ sở dữ liệu tin cậy	
Thư mục chứng chỉ thu hồi	
Thư mục khóa riêng tư	
Vòng khóa công khai	

2. Hãy trả lời các câu hỏi sau:

- *GnuPG* sử dụng loại mật mã nào?

- Hai thành phần chính của mật mã khóa công khai là gì?

- KEY-ID của dấu vân tay khóa công khai 07A6 5898 2D3A F3DD 43E3 DA95 1F3F 3147 FA7F 54C7 là gì?

- Phương pháp nào được sử dụng để phân phối khóa công khai ở cấp độ toàn cục?

3. Hãy thực hiện các bước sau theo đúng thứ tự liên quan đến việc thu hồi khóa riêng tư:

- Cung cấp khóa bị thu hồi cho đối tác của bạn.
- Tạo chứng chỉ thu hồi.
- Nhập chứng chỉ thu hồi vào vòng khóa của bạn.

Thứ tự đúng sẽ là:

Bước 1:	
Bước 2:	
Bước 3:	

4. Về mã hóa tệp, tùy chọn `--armor` có ngụ ý gì trong lệnh `gpg --output Encrypted-message --recipient carol --armor --encrypt unencrypted-message`?

Bài tập Mở rộng

1. Hầu hết các tùy chọn `gpg` đều có cả phiên bản dài và phiên bản ngắn. Hãy hoàn thành bảng sau với phiên bản ngắn tương ứng:

Phiên bản dài	Phiên bản ngắn
<code>--armor</code>	
<code>--output</code>	
<code>--recipient</code>	
<code>--decrypt</code>	
<code>--encrypt</code>	
<code>--sign</code>	

2. Hãy trả lời các câu hỏi sau liên quan đến xuất khoá:

- Bạn sẽ sử dụng lệnh nào để xuất tất cả các khóa công khai của mình sang một tệp có tên `all.key`?

- Bạn sẽ sử dụng lệnh nào để xuất tất cả khóa riêng tư của mình sang một tệp có tên `all_private.key`?

3. Tùy chọn `gpg` nào sẽ cho phép ta thực hiện hầu hết các tác vụ liên quan đến quản lý khóa bằng cách hiển thị một menu?

4. Tùy chọn `gpg` nào sẽ cho phép bạn tạo một chữ ký văn bản thuần túy?

Tóm tắt

Bài học này đã đề cập đến *GNU Privacy Guard* - một lựa chọn tuyệt vời để mã hóa/giải mã và ký số/xác minh các tệp. Chúng ta đã học về:

- cách tạo một cặp khóa.
- cách liệt kê các khóa trong vòng khóa của bạn.
- nội dung của thư mục `~/ .gnupg`.
- `USER-ID` và `KEY-ID` là gì.
- cách phân phối khóa công khai cho đối tác.
- cách phân phối khóa công khai trên phạm vi toàn cục thông qua máy chủ khóa.
- cách thu hồi khóa riêng tư.
- cách mã hóa và giải mã tệp.
- cách ký và xác minh tệp.
- kiến thức cơ bản về *GPG-Agent*.

Các lệnh sau đã được thảo luận trong bài học này:

gpg

Công cụ mã hóa và ký số *OpenPGP*.

Đáp án Bài tập Hướng dẫn

1. Hãy hoàn thành bảng bằng cách cung cấp tên tệp chính xác:

Mô tả	Tên tệp
Cơ sở dữ liệu tin cậy	trustdb.gpg
Thư mục chứng chỉ thu hồi	opengp-revocs.d
Thư mục khóa riêng tư	private-keys-v1.d
Vòng khóa công khai	pubring.kbx

2. Hãy trả lời các câu hỏi sau:

- *GnuPG* sử dụng loại mật mã nào?

Mật mã khóa công khai hoặc mật mã bất đối xứng.

- Hai thành phần chính của mật mã khóa công khai là gì?

Khóa công khai và khóa riêng tư.

- KEY-ID của dấu tay khóa công khai 07A6 5898 2D3A F3DD 43E3 DA95 1F3F 3147 FA7F 54C7 là gì?

FA7F 54C7

- Phương pháp nào được sử dụng để phân phối khóa công khai ở cấp độ toàn cục?

Máy chủ khóa.

3. Hãy thực hiện các bước sau theo đúng thứ tự liên quan đến việc thu hồi khóa riêng tư:

- Cung cấp khóa bị thu hồi cho đối tác của bạn.
- Tạo chứng chỉ thu hồi.
- Nhập chứng chỉ thu hồi vào vòng khóa của bạn.

Thứ tự đúng sẽ là:

Bước 1:	Tạo chứng chỉ thu hồi
Bước 2:	Nhập chứng chỉ thu hồi vào vòng khóa của bạn.

Bước 3:

Cung cấp khóa bị thu hồi cho đối tác của bạn.

- Về mã hóa tệp, tùy chọn `--armor` có ngụ ý gì trong lệnh `gpg --output Encrypted-message --recipient carol --armor --encrypt unencrypted-message?`

Nó tạo ra đầu ra được kết xuất dưới dạng văn bản mã ASCII cho phép bạn sao chép tệp được mã hóa hiện có vào email.

Đáp án Bài tập Mở rộng

1. Hầu hết các tùy chọn `gpg` đều có cả phiên bản dài và phiên bản ngắn. Hãy hoàn thành bảng sau với phiên bản ngắn tương ứng:

Phiên bản dài	Phiên bản ngắn
<code>--armor</code>	<code>-a</code>
<code>--output</code>	<code>-o</code>
<code>--recipient</code>	<code>-r</code>
<code>--decrypt</code>	<code>-d</code>
<code>--encrypt</code>	<code>-e</code>
<code>--sign</code>	<code>-s</code>

2. Hãy trả lời các câu hỏi sau liên quan đến xuất khoá:

- Bạn sẽ sử dụng lệnh nào để xuất tất cả các khóa công khai của mình sang một tệp có tên `all.key`?

```
gpg --export --output all.key hoặc gpg --export -o all.key
```

- Bạn sẽ sử dụng lệnh nào để xuất tất cả khóa riêng tư của mình sang một tệp có tên `all_private.key`?

```
gpg --export-secret-keys --output all_private.key hoặc gpg --export-secret-keys -o all_private.key
```

(`--export-secret-keys` có thể được thay thế bằng `--export-secret-subkeys` với một kết quả hơi khác — hãy kiểm tra man `gpg` để biết thêm thông tin).

3. Tùy chọn `gpg` nào sẽ cho phép ta thực hiện hầu hết các tác vụ liên quan đến quản lý khóa bằng cách hiển thị một menu?

```
--edit-key
```

4. Tùy chọn `gpg` nào sẽ cho phép bạn tạo một chữ ký văn bản thuần túy?

```
--clearsign
```

Ấn bản

© 2024 bởi Linux Professional Institute: Tài liệu Học tập, “LPIC-1 (102) (Phiên bản 5.0)”.

PDF được tạo vào: 2024-02-25

Ấn phẩm này được cấp phép theo Giấy phép Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License (CC BY-NC-ND 4.0). Để xem bản sao của giấy phép này, hãy truy cập

<https://creativecommons.org/licenses/by-nc-nd/4.0/>

Mặc dù Linux Professional Institute đã rất nỗ lực một cách thiện chí để đảm bảo rằng các thông tin và hướng dẫn trong các tài liệu này là chính xác, Linux Professional Institute từ chối mọi trách nhiệm đối với các lỗi hoặc thiếu sót, bao gồm nhưng không giới hạn trách nhiệm đối với thiệt hại do việc sử dụng hoặc phụ thuộc vào tài liệu này. Việc sử dụng các thông tin và hướng dẫn có trong tài liệu này là rủi ro của riêng người dùng. Nếu bất kỳ mẫu mã hoặc công nghệ nào khác mà tài liệu này nhắc tới hoặc mô tả cần tuân theo giấy phép mã nguồn mở hoặc quyền sở hữu trí tuệ của người khác, người dùng có trách nhiệm đảm bảo rằng việc sử dụng của họ tuân thủ các giấy phép và/hoặc quyền đó.

Tài liệu Học tập LPI là một sáng kiến của Linux Professional Institute (<https://lpi.org>). Tài liệu Học tập và các Bản dịch của chúng có thể được tìm thấy tại <https://learning.lpi.org>.

Đối với các câu hỏi và nhận xét về ấn bản này cũng như về toàn bộ dự án, hãy gửi email tới: learning@lpi.org.