



Linux
Professional
Institute

LPIC-1

Versão 5.0
Português

102

Table of Contents

TÓPICO 105: SHELLS E SCRIPTS DO SHELL	1
105.1 Personalizar e trabalhar no ambiente shell	2
105.1 Lição 1	4
Introdução	4
Tipos de shell: Interativo x Não-interativo e Login vs sem login	5
Exercícios Guiados	18
Exercícios Exploratórios	20
Resumo	22
Respostas aos Exercícios Guiados	24
Respostas aos Exercícios Exploratórios	26
105.1 Lição 2	28
Introdução	28
Variáveis: atribuição e referência	28
Variáveis locais ou do Shell	32
Variáveis globais ou de ambiente	35
Exercícios Guiados	45
Exercícios Exploratórios	48
Resumo	50
Respostas aos Exercícios Guiados	52
Respostas aos Exercícios Exploratórios	56
105.1 Lição 3	58
Introdução	58
Criando aliases	58
Criando funções	62
Exercícios Guiados	73
Exercícios Exploratórios	76
Resumo	77
Respostas aos Exercícios Guiados	79
Respostas aos Exercícios Exploratórios	84
105.2 Editar e escrever scripts simples	85
105.2 Lição 1	87
Introdução	87
Estrutura e execução de scripts	88
Variáveis	90
Expressões aritméticas	93
Execução condicional	94
Saída do script	95
Exercícios Guiados	98

Exercícios Exploratórios	99
Resumo	100
Respostas aos Exercícios Guiados	101
Respostas aos Exercícios Exploratórios	102
105.2 Lição 2	103
Introdução	103
Testes ampliados	103
Construções de loop	109
Um exemplo mais elaborado	111
Exercícios Guiados	115
Exercícios Exploratórios	117
Resumo	118
Respostas aos Exercícios Guiados	119
Respostas aos Exercícios Exploratórios	121
TÓPICO 106: INTERFACES DE USUÁRIO E DESKTOPS	122
106.1 Instalar e configurar o X11	123
106.1 Lição 1	124
Introdução	124
A arquitetura do X Window System	125
Configuração do servidor X	128
Wayland	133
Exercícios Guiados	135
Exercícios Exploratórios	136
Resumo	137
Respostas aos Exercícios Guiados	138
Respostas aos Exercícios Exploratórios	139
106.2 Desktops gráficos	140
106.2 Lição 1	141
Introdução	141
X Window System	142
Ambiente de trabalho	142
Ambientes de trabalho populares	144
Interoperabilidade da área de trabalho	146
Acesso não-local	147
Exercícios Guiados	150
Exercícios Exploratórios	151
Resumo	152
Respostas aos Exercícios Guiados	153
Respostas aos Exercícios Exploratórios	154
106.3 Acessibilidade	155

106.3 Lição 1	156
Introdução	156
Configurações de acessibilidade	156
Assistente de teclado e mouse	157
Deficiências visuais	159
Exercícios Guiados	161
Exercícios Exploratórios	162
Resumo	163
Respostas aos Exercícios Guiados	164
Respostas aos Exercícios Exploratórios	165
TÓPICO 107: TAREFAS ADMINISTRATIVAS	166
107.1 Administrar contas de usuário, grupos e arquivos de sistema relacionados	167
107.1 Lição 1	169
Introdução	169
Adicionando contas de usuário	169
Modificando contas de usuário	171
Excluindo contas de usuário	173
Adicionando, modificando e excluindo grupos	173
O diretório de esqueleto	174
O arquivo <code>/etc/login.defs</code>	174
O comando <code>passwd</code>	175
O comando <code>chage</code>	177
Exercícios Guiados	178
Exercícios Exploratórios	179
Resumo	180
Respostas aos Exercícios Guiados	182
Respostas aos Exercícios Exploratórios	184
107.1 Lição 2	186
Introdução	186
<code>/etc/passwd</code>	187
<code>/etc/group</code>	187
<code>/etc/shadow</code>	188
<code>/etc/gshadow</code>	189
Como filtrar os bancos de dados de senha e grupo	189
Exercícios Guiados	191
Exercícios Exploratórios	193
Resumo	194
Respostas aos Exercícios Guiados	195
Respostas aos Exercícios Exploratórios	197
107.2 Automatizar e agendar tarefas administrativas de sistema	199

107.2 Lição 1	201
Introdução	201
Como agendar jobs com o cron	201
Crontabs de usuário	202
Crontabs de sistema	203
Especificações de tempo particulares	204
Variáveis no crontab	204
Criando cron jobs de usuário	205
Criando cron jobs do sistema	206
Configurando o acesso ao agendamento de trabalhos	207
Uma alternativa ao cron	207
Exercícios Guiados	210
Exercícios Exploratórios	212
Resumo	213
Respostas aos Exercícios Guiados	214
Respostas aos Exercícios Exploratórios	216
107.2 Lição 2	218
Introdução	218
Agendamento de trabalhos com at	218
Listar jobs programados com atq	219
Excluir jobs com atrm	220
Configurando o acesso ao agendamento de trabalhos	221
Especificações de tempo	221
Uma alternativa ao at	221
Exercícios Guiados	223
Exercícios Exploratórios	224
Resumo	225
Respostas aos Exercícios Guiados	226
Respostas aos Exercícios Exploratórios	227
107.3 Localização e internacionalização	229
107.3 Lição 1	231
Introdução	231
Fusos horários	232
Horário de verão	236
Idioma e codificação de caracteres	237
Conversão de codificação	240
Exercícios Guiados	241
Exercícios Exploratórios	242
Resumo	243
Respostas aos Exercícios Guiados	244

Respostas aos Exercícios Exploratórios	245
TÓPICO 108: SERVIÇOS ESSENCIAIS DO SISTEMA	246
108.1 Manutenção da data e hora do sistema	247
108.1 Lição 1	249
Introdução	249
Hora local e hora universal	250
Date	250
Relógio do hardware	252
timedatectl	252
Definindo o fuso horário sem timedatectl	254
Configurando data e hora sem timedatectl	255
Exercícios Guiados	258
Exercícios Exploratórios	260
Resumo	261
Respostas aos Exercícios Guiados	263
Respostas aos Exercícios Exploratórios	265
108.1 Lição 2	266
Introdução	266
timedatectl	268
Daemon NTP	269
Configuração do NTP	270
pool.ntp.org	271
ntpdate	271
ntpq	271
chrony	272
Exercícios Guiados	277
Exercícios Exploratórios	279
Resumo	280
Respostas aos Exercícios Guiados	281
Answers to Explorational Exercises	283
108.2 Log do sistema	284
108.2 Lição 1	286
Introdução	286
Logs do sistema	287
Exercícios Guiados	307
Exercícios Exploratórios	309
Resumo	310
Respostas aos Exercícios Guiados	311
108.2 Lição 2	314
Introdução	314

Fundamentos do <code>systemd</code>	314
O diário do sistema: <code>systemd-journald</code>	315
Exercícios Guiados	334
Exercícios Exploratórios	337
Summary	338
Answers to Guided Exercises	339
Answers to Explorational Exercises	342
108.3 Fundamentos de MTA (Mail Transfer Agent)	343
108.3 Lição 1	344
Introdução	344
MTA local e remoto	345
MTAs do Linux	346
O comando <code>mail</code> e Mail User Agents (MUA)	351
Entrega personalizada	352
Exercícios Guiados	355
Exercícios Exploratórios	356
Resumo	357
Respostas aos Exercícios Guiados	358
Respostas aos Exercícios Exploratórios	359
108.4 Configurar impressoras e impressão	360
108.4 Lição 1	361
Introdução	361
O serviço CUPS	362
Instalando uma impressora	366
Gerenciando impressoras	368
Enviando trabalhos de impressão	370
Gerenciando trabalhos de impressão	372
Removendo impressoras	374
Exercícios Guiados	375
Exercícios Exploratórios	376
Resumo	377
Respostas aos Exercícios Guiados	379
Respostas aos Exercícios Exploratórios	380
TÓPICO 109: FUNDAMENTOS DE REDE	382
109.1 Fundamentos de protocolos de internet	383
109.1 Lição 1	384
Introdução	384
IP (Internet Protocol)	384
Exercícios Guiados	393
Exercícios Exploratórios	394

Resumo	395
Respostas aos Exercícios Guiados	396
Respostas aos Exercícios Exploratórios	397
109.1 Lição 2	398
Introdução	398
Transmission Control Protocol (TCP)	400
User Datagram Protocol (UDP)	400
Internet Control Message Protocol (ICMP)	400
IPv6	401
Exercícios Guiados	404
Exercícios Exploratórios	405
Resumo	406
Respostas aos Exercícios Guiados	407
Respostas aos Exercícios Exploratórios	408
109.2 Configuração persistente de rede	409
109.2 Lição 1	410
Introdução	410
A Interface de Rede	410
Nomes de interface	412
Gerenciamento da interface	413
Nomes locais e remotos	415
Exercícios Guiados	420
Exercícios Exploratórios	421
Resumo	422
Respostas aos Exercícios Guiados	423
Respostas aos Exercícios Exploratórios	424
109.2 Lição 2	425
Introdução	425
NetworkManager	425
systemd-networkd	430
Exercícios Guiados	433
Exercícios Exploratórios	434
Resumo	435
Respostas aos Exercícios Guiados	436
Respostas aos Exercícios Exploratórios	437
109.3 Soluções para problemas simples de rede	438
109.3 Lição 1	440
Introdução	440
Sobre o comando ip	441
Máscara de rede e revisão de roteamento	442

Configurando uma interface	443
A tabela de roteamento.....	445
Exercícios Guiados	449
Exercícios Exploratórios.....	450
Resumo	451
Respostas aos Exercícios Guiados	452
Respostas aos Exercícios Exploratórios	454
109.3 Lição 2	456
Introdução.....	456
Testando conexões com ping	456
Traçando Rotas	457
Encontrando MTUs com tracepath	460
Criando Conexões Arbitrárias.....	460
Visualizando conexões atuais e listeners	462
Exercícios Guiados	464
Exercícios Exploratórios.....	465
Resumo	466
Respostas aos Exercícios Guiados	468
Respostas aos Exercícios Exploratórios	470
109.4 Configurar DNS cliente	472
109.4 Lição 1	473
Introdução.....	473
O processo de resolução de nome	473
Classes DNS	474
Ferramentas de resolução de nomes	477
Exercícios Guiados	483
Exercícios Exploratórios.....	484
Resumo	485
Respostas aos Exercícios Guiados	486
Respostas aos Exercícios Exploratórios	487
TÓPICO 110: SEGURANÇA	488
110.1 Tarefas administrativas de segurança	489
110.1 Lição 1	491
Introdução.....	491
Verificando Arquivos com SUID e SGID	491
Gerenciamento e validade das senhas	494
Descobrindo portas abertas	497
Limites em logins de usuário, processos e uso de memória	504
Lidando com usuários conectados	507
Configuração e uso básicos do sudo	510

Exercícios Guiados	515
Exercícios Exploratórios	518
Resumo	519
Respostas aos Exercícios Guiados	521
Respostas aos Exercícios Exploratórios	525
110.2 Configurar a segurança do host	526
110.2 Lição 1	527
Introdução	527
Melhorar a segurança da autenticação com senhas ocultas	527
Como usar um superdaemon para ouvir conexões de rede de entrada	529
Em busca de daemons desnecessários nos serviços	534
Usando TCP wrappers como uma espécie de firewall simples	536
Exercícios Guiados	537
Exercícios Exploratórios	538
Resumo	539
Respostas aos Exercícios Guiados	541
Respostas aos Exercícios Exploratórios	542
110.3 Proteção de dados com criptografia	543
110.3 Lição 1	545
Introdução	545
Configuração e uso básico do cliente OpenSSH	546
O papel das chaves de host do servidor OpenSSH	551
Túneis de porta SSH	553
Exercícios Guiados	558
Exercícios Exploratórios	560
Resumo	561
Respostas aos Exercícios Guiados	562
Respostas aos Exercícios Exploratórios	564
110.3 Lição 2	565
Introdução	565
Como executar a configuração, uso e revogação básicos do GnuPG	565
Como usar o GPG para criptografar, descriptografar, assinar e verificar arquivos	571
Exercícios Guiados	577
Exercícios Exploratórios	579
Resumo	580
Respostas aos Exercícios Guiados	581
Respostas aos Exercícios Exploratórios	583
Imprint	584



Tópico 105: Shells e scripts do Shell



105.1 Personalizar e trabalhar no ambiente shell

Referência ao LPI objectivo

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

Peso

4

Áreas chave de conhecimento

- Definir variáveis de ambiente (por exemplo, PATH) no início da sessão ou quando abrir um novo shell.
- Escrever funções Bash para sequências de comandos frequentemente usadas.
- Manter o esqueleto de diretórios (skeleton) para novas contas de usuários.
- Definir os caminhos de busca de comandos para apontar para os diretórios corretos.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

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

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



105.1 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	105 Shells e scripts do Shell
Objetivo:	105.1 Personalizar e usar o ambiente do shell
Lição:	1 de 3

Introdução

O shell é, sem dúvida, a ferramenta mais poderosa de um sistema Linux. Ele pode ser definido como uma interface entre o usuário e o kernel do sistema operacional. Ele interpreta os comandos inseridos pelo usuário. Assim, todos os administradores de sistema devem saber usar o shell. Como você já deve ter entendido, o Bourne Again Shell (*Bash*) é o shell *de facto* na grande maioria das distribuições Linux.

Uma vez iniciado, a primeira coisa que o Bash — ou qualquer outro shell, aliás — faz é executar uma série de scripts de inicialização. Esses scripts personalizam o ambiente da sessão. Existem scripts que afetam todo o sistema e outros específicos do usuário. Podemos colocar as preferências ou configurações pessoais que melhor atendam às necessidades dos usuários nesses scripts na forma de variáveis, aliases e funções.

A série exata de arquivos de inicialização depende de um parâmetro muito importante: o tipo de shell. Vamos dar uma olhada na variedade existente.

Tipos de shell: Interativo x Não-interativo e Login vs sem login

Para começar, vamos esclarecer os conceitos de *interativo* e *login* no contexto dos shells:

Shells interativos / não-interativos

Este tipo de shell refere-se à interação que ocorre entre o usuário e o shell: O usuário fornece a entrada digitando comandos no terminal com o teclado; o shell fornece a saída imprimindo mensagens na tela.

Shells com login / sem login

Esse tipo de shell se refere ao caso de um usuário que acessa um sistema fornecendo suas credenciais, como nome de usuário e senha.

Os shells interativos e não interativos podem ser de login ou sem login; qualquer combinação possível desses tipos tem seus usos específicos.

Os *shells de login interativos* são executados quando os usuários fazem login no sistema, e são usados para personalizar as configurações de usuário de acordo com suas necessidades. Um bom exemplo do uso desse tipo de shell seria um grupo de usuários pertencentes ao mesmo departamento de uma empresa que precisam de uma determinada variável definida em suas sessões.

Shells sem login interativos são quaisquer outros shells abertos pelo usuário após o login no sistema. Os usuários empregam esses shells durante as sessões para realizar tarefas de manutenção e administrativas, como definir variáveis, data e hora, copiar arquivos, criar scripts, etc.

Por outro lado, os *shells não-interativos* não requerem nenhum tipo de interação humana. Assim, esses shells não pedem entrada de dados pelo usuário e sua saída — quando existe — é, na maioria dos casos, gravada em um log.

Os *shells de login não-interativos* são bastante raros e pouco práticos. Eles praticamente não têm utilidade e nós só estamos falando deles para você conhecer outros aspectos do comportamento do shell. Alguns exemplos incomuns incluem forçar um script a ser executado a partir de um shell de login com `/bin/bash --login <algum_script>` ou canalizando a saída padrão (`stdout`) de um comando para a entrada padrão (`stdin`) de uma conexão ssh:

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

No caso do *shell não-interativo e sem login*, não há interação nem login em nome do usuário; portanto, estamos nos referindo aqui ao uso de scripts automatizados. Esses scripts são usados

principalmente para realizar tarefas administrativas e de manutenção repetitivas, como as incluídas em cronjobs. Nesses casos, o bash não lê nenhum arquivo de inicialização.

Abrindo um terminal

Quando estamos em um ambiente de desktop, podemos abrir um aplicativo de terminal ou mudar para um dos consoles do sistema. Portanto, um novo shell pode ser um shell pts, quando aberto a partir de um emulador de terminal na GUI, ou um shell tty, quando executado em um console do sistema. No primeiro caso, não estamos lidando com um terminal, mas com um emulador de terminal. Por serem parte das sessões gráficas, os emuladores de terminal, como o *gnome-terminal* ou o *konsole*, são muito ricos em recursos e fáceis de usar em comparação com os terminais de interface de usuário baseados em texto. Dentre os emuladores de terminal com menos recursos estão — entre outros — o *XTerm* e o *sakura*.

Usando as combinações `Ctrl + Alt + F1 - F6` podemos ir para os logins do console, que abrem um shell de login interativo baseado em texto. `Ctrl + Alt + F7` leva a sessão de volta para a área de trabalho.

NOTE

`tty` significa teletypewriter (teletipo); `pts` é a abreviação de pseudo terminal slave (pseudo terminal escravo). Para saber mais: `man tty` e `man pts`.

Lançando shells com o bash

Após fazer o login, digite bash em um terminal para abrir um novo shell. Tecnicamente, este shell é um processo filho do shell atual.

Ao iniciar o processo filho bash, podemos especificar diversas opções para definir que tipo de shell queremos iniciar. Eis algumas opções importantes de invocação no bash:

bash -l or bash --login

invoca um shell de login.

bash -i

invoca um shell interativo.

bash --noprofile

com shells de login, ignora o arquivo de inicialização do sistema `/etc/profile` e os arquivos de inicialização em nível de usuário `~/.bash_profile`, `~/.bash_login` e `~/.profile`.

bash --norc

com shells interativos, ignora tanto o arquivo de inicialização do sistema `/etc/bash.bashrc` quanto o arquivo de inicialização em nível de usuário `~/.bashrc`.

bash --rcfile <file>

com shells interativos, considera <file> como arquivo de inicialização, ignorando os arquivos de inicialização do sistema /etc/bash.bashrc e em nível de usuário ~/.bashrc.

Discutiremos abaixo os diversos arquivos de inicialização.

Iniciando shells com su e sudo

Graças a esses dois programas semelhantes, podemos obter tipos específicos de shell:

su

Muda o ID de usuário ou o torna superusuário (root). Com este comando, podemos chamar shells de login e sem login:

- su - user2, su -l user2 ou su --login user2 iniciam um shell de login interativo como user2.
- su user2 inicia um shell sem login interativo como user2.
- su - root ou su - inicia um shell de login interativo como root.
- su root ou su inicia um shell interativo sem login como root.

sudo

Executa comandos como outro usuário (incluindo o superusuário). Como este comando é usado principalmente para obter privilégios de root temporariamente, o usuário que o emprega deve estar no arquivo sudoers. Para adicionar usuários a sudoers, precisamos nos tornar root e então executar:

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

Assim como o su, o sudo permite invocar shells de login e sem login:

- sudo su - user2, sudo su -l user2 ou sudo su --login user2 iniciam um shell de login interativo como user2.
- sudo su user2 inicia um shell sem login interativo como user2.
- sudo -u user2 -s inicia um shell sem login interativo como user2.
- sudo su - root ou sudo su - inicia um shell de login interativo como root.
- sudo -i inicia um shell de login interativo como root.
- sudo -i <algum_comando> inicia um shell de login interativo como root, executa o comando e retorna ao usuário original.

- `sudo su` ou `sudo su` inicia um shell sem login interativo como root.
- `sudo -s` ou `sudo -u root -s` iniciam um shell sem login como root.

Ao usar `su` ou `sudo`, é importante considerar o contexto particular antes de iniciar um novo shell: Precisamos ou não do ambiente do usuário de destino? Se a resposta for sim, usariámos as opções que invocam shells de login; se não, as que invocam shells sem login.

Qual o meu tipo de shell?

Para descobrir em que tipo de shell estamos trabalhando, podemos digitar `echo $0` no terminal e obter a seguinte saída:

Interativo de login

`-bash` ou `-su`

Interativo sem login

`bash` or `/bin/bash`

Não-interativo sem login (scripts)

`<nome_do_script>`

Quantos shells nós temos?

Para ver quantos shells do `bash` estão rodando no sistema, podemos usar o comando `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
```

A usuária `user2` em `debian` se logou em uma sessão GUI (ou X Window System) e abriu *gnome-terminal*, depois pressionou `Ctrl + Alt + F1` para entrar em uma sessão de terminal `tty`. Finalmente, ela retornou à sessão GUI pressionando `Ctrl + Alt + F7` e digitou o comando `ps aux | grep bash`. Assim, a saída mostra um shell sem login interativo por meio do emulador de terminal (`pts/0`) e um shell de login interativo por meio do terminal baseado em texto (`tty1`). Note também como o último campo de cada linha (o comando) é `bash` para o primeiro e `-bash` para o último.

De onde os shells obtêm sua configuração: arquivos de inicialização

Bem, agora que sabemos os tipos de shell que podemos encontrar em um sistema Linux, é hora de aprender quais arquivos de inicialização são executados por qual shell. Observe que os scripts referentes a todo o sistema, ou globais, são postos no diretório `/etc/`, enquanto que os scripts locais ou em nível de usuário são encontrados no diretório inicial do usuário (`~`). Além disso, quando existe mais de um arquivo a ser pesquisado, logo que um é encontrado e executado, os outros são ignorados. Explore e estude esses arquivos com seu editor de texto favorito ou digitando `less <startup_file>`.

NOTE Os arquivos de inicialização podem ser divididos em específicos do Bash (limitados apenas a configurações e comandos do `bash`) e gerais (relacionados à maioria dos shells).

Shell de login interativo

Nível global

`/etc/profile`

Arquivo `.profile` de todo o sistema para o shell Bourne e shells compatíveis com Bourne (incluindo o `bash`). Através de uma série de instruções `if`, esse arquivo define uma série de variáveis como `PATH` e `PS1` conforme necessário, além de buscar e executar — se existirem — o arquivo `/etc/bash.bashrc` e os que estão no diretório `/etc/profile.d`.

`/etc/profile.d/*`

Este diretório pode conter scripts que são executados por `/etc/profile`.

Nível local

`~/.bash_profile`

Este arquivo específico do Bash é usado para configurar o ambiente do usuário. Também pode ser usado para buscar e executar `~/.bash_login` e `~/.profile`.

`~/.bash_login`

Também específico ao Bash, este arquivo só será executado se não houver um arquivo `~/.bash_profile`. Seu nome sugere que deve ser usado para executar comandos necessários no login.

`~/.profile`

Este arquivo não é específico ao Bash e só é originado se nem `~/.bash_profile` nem `~/.bash_login` existirem — o que normalmente é o caso. Assim, a principal finalidade de

`~/.profile` é conferir se um shell Bash está sendo executado e—se estiver—buscar `~/.bashrc` e executá-lo caso exista. Ele normalmente define a variável `PATH` para que ela inclua o diretório privado `~/bin` do usuário se ele existir.

`~/.bash_logout`

Se existir, este arquivo específico do Bash faz algumas operações de limpeza ao sair do shell. Isso pode ser conveniente em certos casos, como as sessões remotas.

Explorando os arquivos de configuração do shell de login interativo

Vamos mostrar alguns desses arquivos em ação modificando `/etc/profile` e `/home/user2/.profile`. A cada um deles, anexamos uma linha para relembrar o arquivo que está sendo executado:

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

NOTE Dois operadores de redirecionamento `>>` acrescentam a saída de um comando a um arquivo existente—sem sobrescrevê-lo. Porém, se o arquivo não existir, ele será criado.

Assim, através da saída de seus respectivos comandos `echo`, sabemos quando cada um desses arquivos é lido e executado. Para demonstrar, vamos ver o que acontece quando `user2` se loga via `ssh` a partir de outra máquina:

```
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
```

Como mostram as duas últimas linhas, deu tudo certo. Além disso, observe três coisas:

- O arquivo global foi executado primeiro.
- Não havia arquivos `.bash_profile` ou `.bash_login` no diretório inicial de `user2`.
- O til (~) expandiu-se para o caminho absoluto do arquivo (`/home/user2/.profile`).

Shell interativo sem login

Nível global

`/etc/bash.bashrc`

Este é o arquivo `.bashrc` de todo o sistema para shells `bash` interativos. Ao longo de sua execução, o `bash` garante que ele esteja sendo executado de maneira interativa, verifica o tamanho da janela após cada comando (atualizando os valores de `LINES` e `COLUMNS` se necessário) e define algumas variáveis.

Nível local

`~/.bashrc`

Além de realizar tarefas semelhantes às descritas para `/etc/bash.bashrc` no nível do usuário (como verificar o tamanho da janela ou se está sendo executado interativamente), este arquivo específico do `Bash` geralmente define algumas variáveis de histórico e busca e executa `~/.bash_aliases`, se existir. Além disso, esse arquivo é normalmente usado para armazenar aliases e funções específicas dos usuários.

Da mesma forma, também é importante notar que `~/.bashrc` é lido se o `bash` detectar que `<stdin>` é uma conexão de rede (como era o caso com a conexão *Secure Shell* (SSH) no exemplo acima).

Explorando arquivos de configuração de shell interativo sem login

Vamos modificar `/etc/bash.bashrc` e `/home/user2/.bashrc`:

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

E isso é o que acontece quando o `user2` inicia um novo shell:

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

Novamente, os dois arquivos foram lidos e executados.

WARNING

Lembre-se, devido à ordem em que os arquivos são executados, os arquivos locais têm precedência sobre os globais.

Shell de login não interativo

Um shell não-interativo com as opções `-l` ou `--login` é forçado a se comportar como um shell de login, e assim os arquivos de inicialização a serem executados serão iguais aos dos shells de login interativos.

Para provar, vamos escrever um script simples e torná-lo executável. Não incluiremos nenhum *shebang* porque invocaremos o *executável do bash* (`/bin/bash` com a opção de login) a partir da linha de comando.

1. Criamos o script `test.sh` contendo a linha `echo 'Hello from a script'` para ser possível provar que o script foi executado com sucesso:

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

2. Tornamos o script executável:

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

3. Finalmente, invocamos `bash` com a opção `-l` para executar o script:

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

Funcionou! Antes de executar o script, o login foi feito e tanto `/etc/profile` quanto `~/.profile` foram executados.

NOTE

Aprenderemos sobre *shebangs* e todos os outros aspectos dos scripts do shell em lições futuras.

Vamos agora mandar a saída padrão (`stdout`) do comando `echo` para a entrada padrão (`stdin`) de uma conexão `ssh` por meio de um pipe (`|`):

```
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
```

Novamente, `/etc/profile` and `~/.profile` são executados. Fora isso, a primeira e a última linha da saída são bastante reveladoras no que diz respeito ao comportamento do shell.

Shell não-interativo sem login

Os scripts não lêem nenhum dos arquivos listados acima, mas procuram a variável de ambiente `BASH_ENV`, expandem seu valor se necessário e o usam como nome de um arquivo de inicialização para ler e executar comandos. Aprenderemos mais sobre *variáveis de ambiente* na próxima lição.

Como dito acima, tipicamente `/etc/profile` e `~/.profile` garantem que `/etc/bash.bashrc` e `~/.bashrc` sejam executados após um login bem-sucedido. A saída do comando a seguir mostra esse fenômeno:

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

Tendo em mente as linhas que anexamos anteriormente aos scripts de inicialização—invocando um shell de login interativo no nível do usuário com `su - user2`—as quatro linhas da saída podem ser explicadas da seguinte forma:

1. `Hello from /etc/bash.bashrc` significa que `/etc/profile` buscou e executou `/etc/bash.bashrc`.
2. `Hello from /etc/profile` significa que `/etc/profile` foi inteiramente lido e executado.

3. Hello from /home/user2/.bashrc significa que ~/.profile buscou e executou ~/.bashrc.
4. Hello from /home/user2/.profile significa que ~/.profile foi inteiramente lido e executado.

Note que, com su - <username> (e também su -l <username> e su --login <username>), garantimos a invocação de um shell de login, ao passo que su <username> teria invocado somente /etc/bash.bashrc e ~/.bashrc.

Encontrar e executar (sourcing)

Nas seções anteriores, discutimos que alguns scripts de inicialização incluem ou executam outros scripts. Esse mecanismo é chamado de *sourcing* e é explicado nesta seção.

Sourcing de arquivos com .

O ponto (.) normalmente é encontrado em arquivos de inicialização.

No arquivo .profile de nosso servidor Debian, podemos encontrar — por exemplo — o seguinte bloco:

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

Já vimos como a execução de um script pode levar à de outro. Assim, a instrução `if` garante que o arquivo \$HOME/.bashrc — se existir (-f) — será lido e executado (source) no login:

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

NOTE

Como veremos na próxima lição, \$HOME é uma variável de ambiente que se expande para o caminho absoluto do diretório inicial do usuário.

Além disso, podemos usar o . sempre que tivermos modificado um arquivo de inicialização e quisermos que as alterações façam efeito sem reinicializar. Por exemplo, podemos:

- adicionar um alias a ~/.bashrc:

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

WARNING

Ao enviar a saída de um comando em um arquivo, lembre-se de não confundir anexar (`>>`) com sobrescrever (`>`).

- exibir a última linha de `~/.bashrc` na saída para verificar se tudo correu bem:

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

NOTE

`!$` se expande para o último argumento do comando anterior, em nosso caso: `~/.bashrc`.

- abrir e executar o arquivo manualmente:

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

- e invocar o alias para demonstrar que funcionou:

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

NOTE

Consulte a próxima lição para aprender sobre *aliases* e *variáveis*.

Sourcing de arquivos com `source`

O comando `source` é sinônimo de `..`. Assim, para buscar e executar `~/.bashrc`, também podemos fazer desta forma:

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

A origem dos arquivos de inicialização do shell: SKEL

`SKEL` é uma variável cujo valor é o caminho absoluto para o diretório `skel`. Esse diretório serve como modelo para a estrutura do sistema de arquivos dos diretórios pessoais dos usuários. Ele inclui os arquivos que serão herdados por qualquer nova conta de usuário criada (incluindo, é claro, os arquivos de configuração dos shells). `SKEL` e outras variáveis relacionadas são armazenadas em `/etc/adduser.conf`, que é o arquivo de configuração para `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 está definido em /etc/skel; assim, os scripts de inicialização que configuram nossos shells estarão lá:

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

WARNING

Lembre-se de que os arquivos que começam com `.` estão ocultos, portanto é preciso usar `ls -a` para vê-los ao listar o conteúdo do diretório.

Vamos agora criar um diretório em /etc/skel para armazenar os scripts pessoais de todos os novos usuários:

1. Como root, entramos em /etc/skel:

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

2. Listamos o conteúdo:

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

3. Criamos o diretório e verificamos se tudo correu como esperado:

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

4. A seguir, removemos user2 junto com seu diretório home:

```
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. Adicionamos user2 novamente para que receba um novo diretório inicial:

```
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. Finalmente, nos logamos como user2 e listamos todos os arquivos em /home/user2 para verificar se tudo correu como esperado:

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

Deu tudo certo.

Exercícios Guiados

1. Analise como os shells foram iniciados na coluna “Shell iniciado com...” e complete com as informações necessárias:

Shell iniciado com...	Interativo?	Login?	Resultado de echo \$0
<code>sudo ssh</code> <code>user2@machine2</code>			
<code>Ctrl + Alt + F2</code>			
<code>su - user2</code>			
<code>gnome-terminal</code>			
Um usuário comum usa o <i>konsole</i> para iniciar uma instância de <i>sakura</i>			
Um script chamado <code>test.sh</code> contendo o comando <code>echo \$0</code>			

2. Escreva os comandos `su` e `sudo` para lançar o shell especificado:

Shell de login interativo como user2

`su:`

`sudo:`

Shell de login interativo como root

`su:`

`sudo:`

Shell sem login interativo como root

`su:`

`sudo:`

Shell sem login interativo como user2**su:****sudo:**

3. Qual arquivo de inicialização é lido quando iniciamos o shell da coluna “Tipo de shell”?

Tipo de shell	/etc/profile	/etc/bash.bash rc	~/.profile	~/.bashrc
Shell de login interativo como user2				
Shell de login interativo como root				
Shell sem login interativo como root				
Shell sem login interativo como user2				

Exercícios Exploratórios

1. No Bash, podemos escrever uma função simples `Hello world!` incluindo o seguinte código em um arquivo vazio:

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

- O que devemos fazer a seguir para disponibilizar a função para o shell?

- Uma vez que ela esteja disponível para o shell atual, como invocá-la?

- Para automatizar o processo, em qual arquivo você colocaria a função e sua invocação para que ela seja executada quando `user2` abrir um terminal de uma sessão do X Window? De que tipo de shell se trata?

- Em qual arquivo você colocaria a função e sua invocação de forma que seja executada quando `root` iniciar um novo shell interativo, seja este um shell de login ou não?

2. Analise o seguinte script básico `Hello world!` do bash:

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

- Suponha que tornemos o script executável e o executemos. Ele seria um script interativo? Por quê?

- O que torna um script interativo?

3. Imagine que você alterou os valores de algumas variáveis em `~/.bashrc` e quer que essas

alterações tenham efeito sem reinicializar. A partir de seu diretório pessoal, como fazer isso de duas maneiras diferentes?

4. John acaba de iniciar uma sessão de X Window em um servidor Linux. Ele abre um emulador de terminal para realizar algumas tarefas administrativas mas, surpreendentemente, a sessão trava e ele precisa abrir um shell de texto.

- Como ele pode abrir um shell `tty`?

- Quais arquivos de inicialização serão abertos e executados (sourcing)?

5. Linda é usuária de um servidor Linux. Ela gentilmente pede ao administrador para lhe atribuir um arquivo `~/.bash_login` para poder imprimir a hora e data na tela quando se logar. Outros usuários gostam da ideia e seguem o exemplo. O administrador tem dificuldades em criar o arquivo para todos os outros usuários no servidor, por isso ele decide implementar uma nova regra e criar `~/.bash_login` para todos os novos usuários em potencial. Como o administrador pode realizar essa tarefa?

Resumo

Nesta lição, aprendemos:

- Os shells definem o ambiente dos usuários em um sistema Linux.
- O *Bash* é o shell mais usado nas distribuições GNU/Linux.
- A primeira tarefa realizada por um shell é a leitura e execução de um ou vários arquivos de inicialização.
- Os conceitos de *interação* e *login* relacionados aos shells.
- Como lançar diferentes tipos de shells com `bash`, `su`, `sudo` e `Ctrl + Alt + F1-F6`.
- Como verificar o tipo de shell com `echo $0`.
- Os arquivos de inicialização locais `~/.bash_profile`, `~/.profile`, `~/.bash_login`, `~/.bash_logout` e `~/.bashrc`.
- Os arquivos de inicialização globais `/etc/profile`, `/etc/profile.d/*`, `/etc/bash.bashrc`.
- Os arquivos locais têm precedência sobre os globais.
- Como redirecionar a saída de um comando com `>` (sobrescrever) e `>>` (anexar).
- O significado do diretório `skel`.
- Como buscar e executar arquivos com sourcing.

Comandos usados nesta lição:

bash

Cria um novo shell.

su

Cria um novo shell.

sudo

Cria um novo shell.

usermod

Modifica uma conta de usuário.

echo

Exibir uma linha de texto.

ps

Exibir um instantâneo dos processos atuais.

less

Um paginador para arquivos longos.

ssh

Inicia uma conexão Open SSH (remotamente).

chmod

Alterar os bits de modo de um arquivo, por exemplo para torná-lo executável.

grep

Imprimir linhas de acordo com um padrão.

ls

Listar o conteúdo do diretório.

cd

Mudar de diretório.

mkdir

Cria um diretório.

deluser

Exclui um usuário.

adduser

Adiciona um novo usuário.

.

Buscar e executar um arquivo.

source

Buscar e executar um arquivo.

tail

Exibe a parte final dos arquivos na saída.

Respostas aos Exercícios Guiados

1. Analise como os shells foram iniciados na coluna “Shell iniciado com...” e complete com as informações necessárias:

Shell iniciado com...	Interativo?	Login?	Resultado de echo \$0
sudo ssh user2@machine2	Sim	Sim	-bash
Ctrl + Alt + F2	Sim	Sim	-bash
su - user2	Sim	Sim	-bash
gnome-terminal	Sim	Não	bash
Um usuário comum usa o <i>konsole</i> para iniciar uma instância de <i>sakura</i>	Sim	Não	/bin/bash
Um script chamado test.sh contendo o comando echo \$0	Não	Não	./test.sh

2. Escreva os comandos su e sudo para lançar o shell especificado:

IShell de login interativo como user2

su

su - user2, su -l user2 ou su --login user2

sudo

sudo su - user2, sudo su -l user2 ou sudo su --login user2

Shell de login interativo como root

su

su - root ou su -

sudo

sudo su - root, sudo su - ou sudo -i

Shell sem login interativo como root**su**`su root ou su`**sudo**`sudo su root, sudo su, sudo -s ou sudo -u root -s`**Shell sem login interativo como user2****su**`su user2`**sudo**`sudo su user2 ou sudo -u user2 -s`

3. Qual arquivo de inicialização é lido quando iniciamos o shell da coluna “Tipo de shell”?

Tipo de shell	/etc/profile	/etc/bash.bashrc	~/.profile	~/.bashrc
Shell de login interativo como user2	Sim	Sim	Sim	Sim
Shell de login interativo como root	Sim	Sim	Não	Não
Shell sem login interativo como root	Não	Sim	Não	Não
Shell sem login interativo como user2	Não	Sim	Não	Sim

Respostas aos Exercícios Exploratórios

1. No Bash, podemos escrever uma função simples `Hello world!` incluindo o seguinte código em um arquivo vazio:

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

- O que devemos fazer a seguir para disponibilizar a função para o shell?

Para tornar a função disponível para o shell atual, é preciso encontrar e executar o arquivo que a inclui.

- Uma vez que ela esteja disponível para o shell atual, como invocá-la?

Ela é invocada digitando seu nome no terminal.

- Para automatizar o processo, em qual arquivo você colocaria a função e sua invocação para que ela seja executada quando `user2` abrir um terminal de uma sessão do X Window? De que tipo de shell se trata?

O melhor arquivo para colocá-la é `/home/user2/.bashrc`. O shell invocado seria um shell interativo sem login.

- Em qual arquivo você colocaria a função e sua invocação de forma que seja executada quando root iniciar um novo shell interativo, seja este um shell de login ou não?

Em `/etc/bash.bashrc`, já que este arquivo será executado em todos os shells interativos — sejam de login ou não.

2. Analise o seguinte script básico `Hello world!` do bash:

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

- Suponha que tornemos o script executável e o executemos. Ele seria um script interativo? Por quê?

Não, pois não há interação humana e nem comandos digitados pelo usuário.

- O que torna um script interativo?

O fato de que requer entrada de dados pelo usuário.

- Imagine que você alterou os valores de algumas variáveis em `~/.bashrc` e quer que essas alterações tenham efeito sem reiniciar. A partir de seu diretório pessoal, como fazer isso de duas maneiras diferentes?

```
$ source .bashrc
```

ou

```
$ . .bashrc
```

- John acaba de iniciar uma sessão de X Window em um servidor Linux. Ele abre um emulador de terminal para realizar algumas tarefas administrativas mas, surpreendentemente, a sessão trava e ele precisa abrir um shell de texto.

- Como ele pode abrir um shell `tty`?

Ele poderia pressionar `Ctrl + Alt + F1 - F6` para entrar em um dos seis shells `tty`.

- Quais arquivos de inicialização serão abertos e executados (sourcing)?

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

- Linda é usuária de um servidor Linux. Ela gentilmente pede ao administrador para lhe atribuir um arquivo `~/.bash_login` para poder imprimir a hora e data na tela quando se logar. Outros usuários gostam da ideia e seguem o exemplo. O administrador tem dificuldades em criar o arquivo para todos os outros usuários no servidor, por isso ele decide implementar uma nova regra e criar `~/.bash_login` para todos os novos usuários em potencial. Como o administrador pode realizar essa tarefa?

Para isso, ele poderia colocar `.bash_login` no diretório `/etc/skel`.



105.1 Lição 2

Certificação:	LPIC-1
Versão:	5.0
Tópico:	105 Shells e scripts do Shell
Objetivo:	105.1 Personalizar e usar o ambiente do shell
Lição:	2 de 3

Introdução

Pense em uma variável como uma caixa imaginária na qual colocamos temporariamente uma informação. Como no caso dos scripts de inicialização, o Bash classifica as variáveis como *shell/local* (aqueles que vivem apenas dentro dos limites do shell em que foram criadas) ou *ambiente/global* (aqueles que são herdadas por shells e/ou processos filhos). Na lição anterior, conhecemos melhor os shells e seus scripts de configuração ou inicialização. Agora é o momento de frisar que o poder desses arquivos de inicialização está no fato de que eles nos permitem usar variáveis—bem como aliases e funções—que nos ajudam a criar e personalizar o ambiente de shell de nossa escolha.

Variáveis: atribuição e referência

Uma variável pode ser definida como um nome que contém um valor.

No Bash, a atribuição um valor a um nome é chamado de *atribuição de variável* e é a maneira pela qual criamos ou definimos variáveis. Por outro lado, o processo de acesso ao valor contido no nome é denominado *referenciamento de variável*.

A sintaxe para atribuir variáveis é:

```
<nome_da_variável>=<valor_da_variável>
```

Por exemplo:

```
$ distro=zorinos
```

A variável `distro` é igual a `zorinos`, ou seja, há uma parte da memória que contém o valor `zorinos` — sendo `distro` o indicador para ele.

Observe, no entanto, que não pode haver nenhum espaço em qualquer lado do sinal de igual ao se atribuir uma variável:

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

Devido ao nosso erro, o Bash leu `distro` e `zorinos` como comandos.

Para referenciar uma variável (ou seja, para verificar seu valor), usamos o comando `echo` antes do nome da variável com um sinal `$`:

```
$ echo $distro
zorinos
```

Nomes de variáveis

Ao escolher o nome das variáveis, existem certas regras que devemos levar em consideração.

O nome da variável pode conter letras (a-z,A-Z), números (0-9) e sublinhados (_):

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

```
$ distro_1=zorinos
$ echo $distro_1
zorinos
$ _distro=zorinos
$ echo $_distro
zorinos
```

Ele não deve começar com um número para não confundir o Bash:

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

Não deve conter espaços (nem mesmo entre aspas); por convenção, os sublinhados são usados no lugar dos espaços:

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

Valores das variáveis

No que diz respeito à referência ou valor das variáveis, também é importante considerar uma série de regras.

As variáveis podem conter quaisquer caracteres alfanuméricos (a-z,A-Z,0-9), além da maioria dos outros caracteres (?,!,*.,/, etc.):

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

Os valores das variáveis devem ser postos entre aspas se contiverem espaços simples:

```
$ 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
```

Os valores das variáveis também devem ser postos entre aspas se contiverem caracteres como os usados para redirecionamento (<,>) ou o símbolo de barra vertical (|). A única coisa que o comando a seguir faz é criar um arquivo vazio chamado zorin:

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

$ ls zorin
zorin
```

Mas quando usamos as aspas, a coisa funciona:

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

No entanto, aspas simples e duplas nem sempre são intercambiáveis. Dependendo do que estamos fazendo com uma variável (atribuindo ou referenciando), o uso de uma ou de outra tem implicações e produzirá resultados diferentes. No contexto da atribuição de variáveis, as aspas simples consideram *literalmente* todos os caracteres do valor da variável, enquanto as aspas duplas permitem a substituição de variáveis:

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

Por outro lado, ao referenciar uma variável cujo valor inclui alguns espaços iniciais (ou extras)—às vezes combinados com asteriscos—é obrigatório usar aspas duplas após o comando echo para evitar *divisão de campos* e *expansão de nome de caminho*:

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

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

Se a referência da variável contiver um ponto de exclamação no final, este deve ser o último caractere da string (caso contrário, o Bash pensará que estamos nos referindo a um evento de history):

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

Todas as barras invertidas devem ser escapadas com outra barra invertida. Aliás, se uma barra invertida for o último caractere na string e não o escaparmos, o Bash interpretará que queremos uma quebra de linha e criará uma nova linha:

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

Nas próximas duas seções, resumiremos as principais diferenças entre as variáveis *locais* e *de ambiente*.

Variáveis locais ou do Shell

Variáveis locais ou de shell existem apenas no shell em que foram criadas. Por convenção, as variáveis locais são escritas em letras minúsculas.

Para realizar alguns testes, vamos criar uma variável local. Conforme explicado acima, escolhemos um nome de variável apropriado e o igualamos a um valor apropriado. Por exemplo:

```
$ reptile=tortoise
```

Vamos agora usar o comando `echo` para referenciar nossa variável e verificar se tudo correu

conforme o esperado:

```
$ echo $reptile
tortoise
```

Em certos cenários—como ao escrever scripts—a imutabilidade pode ser um recurso interessante das variáveis. Se quisermos que nossas variáveis sejam imutáveis, podemos criá-las em modo `readonly` (somente leitura):

```
$ readonly reptile=tortoise
```

Ou transformá-las em somente leitura depois de criá-las:

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

Agora, se tentarmos alterar o valor de `reptile`, o Bash se recusará:

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

NOTE

Para listar todas as variáveis somente leitura da sessão atual, digite `readonly` ou `readonly -p` no terminal.

Um comando útil ao lidar com variáveis locais é `set`.

O `set` exibe uma saída com todas as variáveis e funções do shell atribuídas atualmente. Como podem ser muitas linhas (experimente você mesmo!), recomendamos usá-lo em combinação com um paginador como o `less`:

```
$ set | less
BASH=/bin/bash
BASHOPTS=checkwinsize:cmdhist:complete_fullquote:expand_aliases:extglob:extquote:force_fignore: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'
( . . . )
```

Será que nossa variável `reptile` está aqui?

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

Sim, aí está ela!

Todavia, `reptile` — por ser uma variável local — não será herdada por nenhum processo filho nascido do shell atual:

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

E, é claro, também não podemos ecoar seu valor:

```
$ echo $reptile
$
```

NOTE Ao digitar o comando `bash` no terminal, abrimos um novo shell (filho).

Para remover variáveis definidas (locais ou globais), usamos o comando `unset`:

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

NOTE `unset` deve ser seguido somente pelo nome da variável (não precedido pelo símbolo `$`).

Variáveis globais ou de ambiente

Existem variáveis globais ou de ambiente para o shell atual, bem como para todos os processos subsequentes gerados a partir dele. Por convenção, as variáveis de ambiente são escritas em letras maiúsculas:

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

Podemos passar recursivamente o valor dessas variáveis para outras variáveis e o valor da última acabará por se expandir para o da primeira:

```
$ 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
```

Para que uma variável local do shell se torne uma variável de ambiente, usamos o comando `export`:

```
$ export reptile
```

Com `export reptile`, transformamos nossa variável local em uma variável de ambiente para que os shells filhos possam reconhecê-la e usá-la:

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

Da mesma maneira, `export` pode ser usado para definir e exportar uma variável de uma só vez:

```
$ export amphibian=frog
```

Agora podemos abrir uma nova instância do Bash e referenciar com sucesso a nova variável:

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

NOTE

Com `export -n <VARIABLE-NAME>`, a variável será novamente transformada em variável local do shell.

O comando `export` também nos dá uma lista de todas as variáveis de ambiente existentes quando inserido sozinho (ou com a opção `-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

O comando `declare -x` equivale a `export`.

Dois outros comandos que podem ser usados para imprimir na tela uma lista de todas as variáveis de ambiente são `env` e `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
```

Além de ser um sinônimo de env, às vezes podemos usar printenv de forma semelhante à que usamos o comando echo para verificar o valor de uma variável:

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

Note, entretanto, que com printenv o nome da variável não é precedido por \$.

NOTE

PWD armazena o caminho do diretório de trabalho atual. Aprenderemos mais sobre esta e outras variáveis de ambiente comuns posteriormente.

Executando um programa em um ambiente modificado

O env pode ser usado para modificar o ambiente do shell no momento da execução de um programa.

Para iniciar uma nova sessão do Bash com o ambiente o mais vazio possível — removendo a maioria das variáveis (além de funções e aliases) — usamos env com a opção -i:

```
$ env -i bash
```

Agora, a maioria das nossas variáveis de ambiente se foi:

```
$ echo $USER
$
```

Restam apenas algumas:

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

Também podemos usar `env` para definir uma variável particular para um programa determinado.

Em nossa lição anterior, ao discutir *shells não interativos sem login*, vimos como os scripts não leem nenhum arquivo de inicialização padrão, mas, em vez disso, procuram o valor da variável `BASH_ENV` e a usam como arquivo de inicialização, se ela existir.

Vamos demonstrar esse processo:

1. Criamos nosso próprio arquivo de inicialização chamado `.startup_script` com o seguinte conteúdo:

```
CROCODILIAN=caiman
```

2. Escrevemos um script do Bash chamado `test_env.sh` com o seguinte conteúdo:

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

3. Definimos o bit executável para nosso script `test_env.sh`:

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

4. Finalmente, usamos `env` para fazer com que `BASH_ENV` rode `.startup_script` para `test_env.sh`:

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

O comando `env` está implícito, mesmo que o eliminemos:

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

NOTE

Se você não entendeu a linha `#!/bin/bash` ou o comando `chmod +x`, não entre em pânico! Aprenderemos todo o necessário sobre os scripts do shell em lições futuras. Neste momento, lembre-se apenas de que para executar um script de dentro de seu próprio diretório, usamos `./some_script`.

Variáveis de ambiente comuns

Agora é hora de revisar algumas das variáveis de ambiente mais relevantes que são definidas nos arquivos de configuração do Bash.

DISPLAY

Relacionada ao servidor X, o valor desta variável geralmente se compõe de três elementos:

- O nome do host (`localhost` se esse dado estiver ausente) no qual o servidor X está rodando.
- Dois pontos como delimitador.
- Um número (normalmente é `0` e se refere à tela do computador).

```
$ printenv DISPLAY
:0
```

Um valor vazio para esta variável indica um servidor sem um X Window System. Um número extra — como em `my.xserver:0:1` — faria referência ao número da tela, caso haja mais de uma.

HISTCONTROL

Esta variável controla quais comandos são salvos em `HISTFILE` (veja abaixo). São três valores possíveis:

ignorespace

Comandos iniciados com um espaço não serão salvos.

ignoredups

Um comando idêntico ao anterior não será salvo.

ignoreboth

Comandos que se encaixam em uma das duas categorias anteriores não serão salvos.

```
$ echo $HISTCONTROL  
ignoreboth
```

HISTSIZE

Define o número de comandos a armazenar na memória enquanto durar a sessão do shell.

```
$ echo $HISTSIZE  
1000
```

HISTFILESIZE

Define o número de comandos a serem salvos em HISTFILE, tanto no início quanto no final da sessão:

```
$ echo $HISTFILESIZE  
2000
```

HISTFILE

O nome do arquivo que armazena todos os comandos à medida que são digitados. Por padrão, esse arquivo está localizado em `~/ .bash_history`:

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

NOTE

Para visualizar o conteúdo de HISTFILE, é só digitar `history`. Alternativamente, podemos especificar o número de comandos que queremos ver passando um argumento (o número dos comandos mais recentes) para `history`, como em `history 3`.

HOME

Esta variável armazena o caminho absoluto do diretório home do usuário atual e é definido quando o usuário faz login.

Este trecho de código—de `~/.profile`—é auto-explicativo (ele abre e executa `"$HOME/.bashrc"`, se existir):

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

fi

NOTE Se você não entendeu muito bem a declaração **if**, não se preocupe: basta consultar as lições sobre os scripts do shell.

Lembre que `~` equivale a `$HOME`:

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

NOTE Os comandos podem ser concatenados com um ponto e vírgula (`;`).

Também podemos provar esse ponto com uma declaração **if**:

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

NOTE Lembre-se: o sinal de igual `=` é usado para a atribuição de variáveis. `==` é usado para testar a igualdade.

HOSTNAME

Esta variável armazena o nome TCP/IP do computador hospedeiro:

```
$ echo $HOSTNAME
debian
```

HOSTTYPE

Armazena a arquitetura do processador do computador hospedeiro:

```
$ echo $HOSTTYPE
x86_64
```

LANG

Esta variável salva o idioma do sistema:

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

LD_LIBRARY_PATH

Esta variável consiste em um conjunto de diretórios separados por dois pontos nos quais as bibliotecas compartilhadas são compartilhadas por programas:

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

MAIL

Esta variável armazena o arquivo no qual o Bash procura por email:

```
$ echo $MAIL  
/var/mail/carol
```

Outro valor comum para esta variável é /var/spool/mail/\$USER.

MAILCHECK

Esta variável armazena um valor numérico que indica, em segundos, a frequência com que o Bash verifica se há novas mensagens:

```
$ echo $MAILCHECK  
60
```

PATH

Esta variável de ambiente armazena a lista de diretórios onde o Bash procura por arquivos executáveis quando instruído a executar qualquer programa. Em nossa máquina de exemplo, esta variável é definida através do arquivo de sistema /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
```

Através da declaração `if`, a identidade do usuário é testada e—dependendo do resultado do teste (`root` ou outro)—obtemos um `PATH` ou o outro. Finalmente, o `PATH` escolhido é propagado com `export`.

Há duas coisas a observar em relação ao valor de PATH:

- Os nomes de diretórios são escritos usando caminhos absolutos.
- O sinal de dois pontos é usado como delimitador.

Se quisermos incluir a pasta `/usr/local/sbin` no `PATH` para usuários regulares, modificamos a linha para que fique assim:

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

Agora podemos ver como o valor da variável muda quando fazemos login como usuário regular:

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

NOTE Também poderíamos ter adicionado `/usr/local/sbin` ao `PATH` do usuário na linha de comando, seja digitando `PATH=/usr/local/sbin:$PATH` ou `PATH=$PATH:/usr/local/sbin`—a primeira opção torna `/usr/local/sbin` o primeiro diretório em que os arquivos executáveis serão buscados; a segunda faz dele o último.

PS1

Essa variável armazena o valor do prompt do Bash. No trecho de código a seguir (igualmente de `/etc/profile`), a declaração `if` testa a identidade do usuário e lhe atribui um prompt bastante personalizado (# para root ou \$ para usuários regulares):

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

NOTE O id de root é 0. Entre como root e teste você mesmo com `id -u`.

Eis algumas outras variáveis de prompt:

PS2

Normalmente definido como > e usado como prompt de continuação para comandos longos de muitas linhas.

PS3

Usado como prompt para o comando select.

PS4

Normalmente definido como + e usado para depuração.

SHELL

Esta variável armazena o caminho absoluto do shell atual:

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

USER

Armazena o nome do usuário atual:

```
$ echo $USER  
carol
```

Exercícios Guiados

1. Analise a atribuição de variáveis na coluna “Comando(s)” e indique se a variável resultante é “Local” ou “Global”:

Comando(s)	Local	Global
debian=mother		
ubuntu=deb-based		
mint=ubuntu-based; export mint		
export suse=rpm-based		
zorin=ubuntu-based		

2. Estude o “Comando” e a “Saída” e explique o significado:

Comando	Saída	Significado
echo \$HISTCONTROL	ignoreboth	
echo ~	/home/carol	
echo \$DISPLAY	reptilium:0:2	
echo \$MAILCHECK	60	
echo \$HISTFILE	/home/carol/.bash_histoy	

3. As variáveis da coluna “Comando errado” estão definidas incorretamente. Forneça as informações que faltam em “Comando correto” e “Referência da variável” para obter a “Saída esperada”:

Comando errado	Comando correto	Referência da variável	Saída esperada
lizard =chameleon			chameleon
cool lizard=chameleon			chameleon
lizard=cha me leon			cha me leon

Comando errado	Comando correto	Referência da variável	Saída esperada
lizard=/** chameleon **/			/** chameleon **/
win_path=C:\path\t o\dir\			C:\path\to\dir\

4. Considere a finalidade e escreva o comando apropriado:

Finalidade	Comando
Definir o idioma do shell atual para Espanhol UTF-8 (es_ES.UTF-8).	
Imprimir o nome do diretório de trabalho atual.	
Referenciar a variável de ambiente que armazena as informações sobre as conexões ssh.	
Definir PATH de forma a incluir /home/carol/scripts como o último diretório para pesquisar executáveis.	
Definir o valor de my_path como PATH.	
Definir o valor de my_path para o de PATH.	

5. Crie uma variável local chamada `mammal` e atribua a ela o valor `gnu`:

6. Usando a substituição de variáveis, crie outra variável local chamada `var_sub` com o valor apropriado para que, ao referenciá-la com `echo $var_sub`, possamos obter: `The value of mammal is gnu`:

7. Transforme `mammal` em variável de ambiente:

8. Procure por ela com `set` e `grep`:

9. Procure por ela com `env` e `grep`:

10. Crie, em dois comandos consecutivos, uma variável de ambiente chamada `BIRD` cujo valor é `penguin`:

11. Crie, em um único comando, uma variável de ambiente chamada `NEW_BIRD` cujo valor é `yellow-eyed penguin`:

12. Supondo que você é `user2`, crie uma pasta chamada `bin` em seu diretório inicial:

13. Digite o comando para adicionar a pasta `~/bin` a seu `PATH` para que este seja o primeiro diretório em que o bash busca por binários:

14. Para garantir que o valor de `PATH` permaneça inalterado após uma reinicialização, qual trecho de código — na forma de uma declaração `if` — você incluiria em `~/.profile`?

Exercícios Exploratórios

1. let: mais do que avaliação de expressão aritmética:

- Busque por `let` na página de manual ou na internet para entender suas implicações ao definir variáveis e crie uma nova variável local chamada `my_val` cujo valor é 10 — como resultado da soma de 5 e 5:

- A seguir, crie outra variável chamada `your_val` cujo valor é 5 — como resultado da divisão do valor de `my_val` por 2:

2. O resultado de um comando em uma variável? Pois é, isso é possível: chamamos de *substituição de comando*. Explore e estude a seguinte função, chamada `music_info`:

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

O resultado do comando `ls -l1t ~/Music | head -n 6` se torna o valor da variável `latest_music`. Daí a variável `latest_music` é referenciada no comando `echo` (cuja saída mostra o número total de bytes ocupados pela pasta `Music` e os cinco últimos arquivos de música armazenados na pasta `Music` — um por linha).

Qual dos seguintes é um sinônimo válido para

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

Opção A:

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

Opção B:

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

Opção C:

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

Resumo

Nesta lição, aprendemos:

- As variáveis são uma parte importantíssima do ambiente do shell, pois são usadas pelo próprio shell e também por outros programas.
- Como atribuir e referenciar variáveis.
- As diferenças entre variáveis *locais* e *globais* (ou *de ambiente*).
- Como tornar as variáveis *somente leitura*.
- Como transformar uma variável local em uma variável de ambiente com o comando `export`.
- Como listar todas as variáveis de ambiente.
- Como executar um programa em um ambiente modificado.
- Como tornar as variáveis persistentes com a ajuda de scripts de inicialização.
- Algumas variáveis de ambiente comuns: `DISPLAY`, `HISTCONTROL`, `HISTSIZE`, `HISTFILESIZE`, `HISTFILE`, `HOME`, `HOSTNAME`, `HOSTTYPE`, `LANG`, `LD_LIBRARY_PATH`, `MAIL`, `MAILCHECK`, `PATH`, `PS1` (e outras variáveis de prompt), `SHELL` e `USER`.
- O significado do til (~).
- Noções básicas das declarações `if`.

Comandos usados nesta lição:

`echo`

Referencia uma variável.

`ls`

Listar o conteúdo do diretório.

`readonly`

Torna as variáveis imutáveis. Lista todas as variáveis somente leitura na sessão atual.

`set`

Lista todas as variáveis e funções na sessão atual.

`grep`

Imprimir linhas de acordo com um padrão.

bash

Lança um novo shell

unset

Remove as variáveis definidas.

export

Transforma uma variável local em uma variável de ambiente. Lista as variáveis de ambiente.

env

Lista as variáveis de ambiente. Executa um programa em um ambiente modificado.

printenv

Lista as variáveis de ambiente. Referencia uma variável.

chmod

Alterar os bits de modo de um arquivo, por exemplo para torná-lo executável.

history

Lista os comandos anteriores.

su

Alterar o ID do usuário ou torná-lo superusuário.

id

Imprime o ID do usuário.

Respostas aos Exercícios Guiados

1. Analise a atribuição de variáveis na coluna “Comando(s)” e indique se a variável resultante é “Local” ou “Global”:

Comando(s)	Local	Global
debian=mother	Sim	Não
ubuntu=deb-based	Sim	Não
mint=ubuntu-based; export mint	Não	Sim
export suse=rpm-based	Não	Sim
zorin=ubuntu-based	Sim	Não

2. Estude o “Comando” e a “Saída” e explique o significado:

Comando	Saída	Significado
echo \$HISTCONTROL	ignoreboth	Tanto os comandos duplicados quanto aqueles que começam com um espaço não serão salvos em history.
echo ~	/home/carol	O HOME de carol é /home/carol.
echo \$DISPLAY	reptilium:0:2	a máquina reptilium tem um servidor X rodando e estamos usando a segunda tela do sistema.
echo \$MAILCHECK	60	Verificação do email a cada um minuto.
echo \$HISTFILE	/home/carol/.bash_history	history será salvo em /home/carol/.bash_history.

3. As variáveis da coluna “Comando errado” estão definidas incorretamente. Forneça as informações que faltam em “Comando correto” e “Referência da variável” para obter a “Saída esperada”:

Comando errado	Comando correto	Referência da variável	Saída esperada
lizard =chameleon	lizard=chameleon	echo \$lizard	chameleon
cool lizard=chameleon	cool_lizard=chamel eon (por exemplo)	echo \$cool_lizard	chameleon
lizard=cha me leon	lizard="cha me leo n" ou lizard='cha me leo n'	echo \$lizard	cha me leon
lizard=/** chameleon **/	lizard="/** chameleon **/" ou lizard='/** chameleon **/'	echo "\$lizard"	/** chameleon **/
win_path=C:\path\t o\dir\	win_path=C:\\path\\ \\to\\dir\\	echo \$win_path	C:\\path\\to\\dir\\

4. Considere a finalidade e escreva o comando apropriado:

Finalidade	Comando
Definir o idioma do shell atual para Espanhol UTF-8 (es_ES.UTF-8).	LANG=es_ES.UTF-8
Imprimir o nome do diretório de trabalho atual.	echo \$PWD ou pwd
Referenciar a variável de ambiente que armazena as informações sobre as conexões ssh.	echo \$SSH_CONNECTION
Definir PATH de forma a incluir /home/carol/scripts como o último diretório para pesquisar executáveis.	PATH=\$PATH:/home/carol/scripts
Definir o valor de my_path como PATH.	my_path=PATH
Definir o valor de my_path para o de PATH.	my_path=\$PATH

5. Crie uma variável local chamada `mammal` e atribua a ela o valor `gnu`:

```
mammal=gnu
```

6. Usando a substituição de variáveis, crie outra variável local chamada `var_sub` com o valor apropriado para que, ao referenciá-la com `echo $var_sub`, possamos obter: `The value of mammal is gnu`:

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

7. Transforme `mammal` em variável de ambiente:

```
export mammal
```

8. Procure por ela com `set` e `grep`:

```
set | grep mammal
```

9. Procure por ela com `env` e `grep`:

```
env | grep mammal
```

10. Crie, em dois comandos consecutivos, uma variável de ambiente chamada `BIRD` cujo valor é `penguin`:

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

11. Crie, em um único comando, uma variável de ambiente chamada `NEW_BIRD` cujo valor é `yellow-eyed penguin`:

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

ou

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

12. Supondo que você é `user2`, crie uma pasta chamada `bin` em seu diretório inicial:

```
mkdir ~ /bin
```

ou

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

ou

```
mkdir $HOME/bin
```

13. Digite o comando para adicionar a pasta `~/bin` a seu `PATH` para que este seja o primeiro diretório em que o bash busca por binários:

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

`PATH=~/bin:$PATH` ou `PATH=/home/user2/bin:$PATH` são igualmente válidos.

14. Para garantir que o valor de `PATH` permaneça inalterado após uma reinicialização, qual trecho de código — na forma de uma declaração `if` — você incluiria em `~/.profile`?

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

Respostas aos Exercícios Exploratórios

1. let: mais do que avaliação de expressão aritmética:

- Busque por `let` na página de manual ou na internet para entender suas implicações ao definir variáveis e crie uma nova variável local chamada `my_val` cujo valor é 10 — como resultado da soma de 5 e 5:

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

ou

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

- A seguir, crie outra variável chamada `your_val` cujo valor é 5 — como resultado da divisão do valor de `my_val` por 2:

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

ou

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

2. O resultado de um comando em uma variável? Pois é, isso é possível: chamamos de *substituição de comando*. Explore e estude a seguinte função, chamada `music_info`:

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

O resultado do comando `ls -l1t ~/Music | head -n 6` se torna o valor da variável `latest_music`. Daí a variável `latest_music` é referenciada no comando `echo` (cuja saída mostra o número total de bytes ocupados pela pasta `Music` e os cinco últimos arquivos de música armazenados na pasta `Music` — um por linha).

Qual dos seguintes é um sinônimo válido para

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

É a opção A:

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



105.1 Lição 3

Certificação:	LPIC-1
Versão:	5.0
Tópico:	105 Shells e scripts do Shell
Objetivo:	105.1 Personalizar e usar o ambiente do shell
Lição:	3 de 3

Introdução

Agora que aprendemos sobre shells, scripts de inicialização e variáveis nas lições anteriores, vamos concluir o tópico da personalização do shell apresentando dois elementos de shell bastante interessantes: *aliases* e *funções*. Na verdade, o que compõe o ambiente do shell é todo o grupo de variáveis, aliases e funções — e a maneira como se influenciam mutuamente.

Esses dois recursos do shell são flexíveis e ajudam a ganhar tempo no trabalho. Seu ponto forte está no conceito de encapsulamento: eles oferecem a possibilidade de reunir — sob um único comando — uma série de comandos repetitivos ou recorrentes.

Criando aliases

Um alias é um nome substituto para outro(s) comando(s). Ele pode ser executado como um comando normal, mas, na verdade, executa outro comando de acordo com a definição do alias.

A sintaxe para declarar aliases é bastante simples. Os aliases são declarados escrevendo-se a palavra-chave `alias` seguida pela atribuição do alias. Por sua vez, a atribuição do alias consiste no *nome do alias*, um *sinal de igual* e um ou mais *comandos*:

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

Por exemplo:

```
$ alias oldshell=sh
```

Este alias bizarro inicia uma instância do shell `sh` original quando o usuário digitar `oldshell` no terminal:

```
$ oldshell
$
```

O poder dos aliases reside no fato de nos permitirem escrever versões curtas de comandos longos:

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

NOTE Para saber mais sobre o `ls` e suas cores, digite `man dir_colors` no terminal.

Da mesma forma, podemos criar aliases para uma série de comandos concatenados — o ponto e vírgula (`;`) é usado como um delimitador. Podemos, por exemplo, ter um alias que nos dá informações sobre a localização do executável `git` e sua versão:

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

Para invocar um alias, digitamos seu nome no terminal:

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

O comando `alias` produz uma lista de todos os aliases disponíveis no sistema:

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

O comando `unalias` remove aliases. Podemos, por exemplo, digitar `unalias git-info` e ver como ele desaparece da listagem:

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

Como vimos com `alias hi='echo We salute you.'` em uma lição anterior, precisamos colocar os comandos entre aspas (simples ou duplas) quando—devido aos argumentos ou parâmetros—eles contêm espaços :

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

Os comandos que incluem opções também têm espaços:

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

Agora, `ll` listará todos os arquivos — incluindo os ocultos (`a`) — no formato longo (`l`).

Podemos referenciar variáveis em aliases:

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

A variável também pode ser atribuída dentro do alias:

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

Podemos escapar um alias com `\`:

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

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

O escape de um alias é útil quando um alias tem o mesmo nome de um comando regular. Nesse caso, o alias tem precedência sobre o comando original, que, no entanto, ainda pode ser acessado escapando-se o alias.

Da mesma forma, podemos colocar um alias dentro de outro alias:

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

Além disso, também é possível colocar uma função dentro de um alias, como será demonstrado mais adiante.

Expansão e avaliação de aspas em aliases

Ao se usar aspas com variáveis de ambiente, as aspas simples tornam a expansão dinâmica:

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

No entanto, com aspas duplas, a expansão é feita estaticamente:

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

Persistência de aliases: scripts de inicialização

Como no caso das variáveis, para que nossos aliases se tornem persistentes devemos colocá-los em scripts de inicialização que são originados quando o sistema é iniciado. Como já sabemos, um bom arquivo para os usuários colocarem seus aliases pessoais é `~/.bashrc`. Já deve haver alguns aliases por lá (a maioria deles comentados e prontos para uso, bastando remover o `#` inicial):

```
$ 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
```

Como podemos ver nas últimas três linhas, é possível ter nosso próprio arquivo dedicado aos aliases—`~/.bash_aliases`—para o `.bashrc` abrir e executar a cada inicialização do sistema. Ao escolher essa opção, criamos e preenchemos esse arquivo:

```
#####
# .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'
```

Criando funções

Comparadas aos aliases, as funções são mais programáticas e flexíveis, especialmente quando se trata de aproveitar todo o potencial das variáveis internas especiais e parâmetros posicionais do *Bash*. Elas também são ótimas para trabalhar com estruturas de controle de fluxo, como loops ou

condicionais. Podemos pensar em uma função como um comando que inclui lógica por meio de blocos ou coleções de outros comandos.

Duas sintaxes para criar funções

Existem duas sintaxes válidas para definir funções.

Usando a palavra-chave `function`

Por um lado, podemos usar a palavra-chave `function`, seguida pelo nome da função e os comandos entre chaves:

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

Using ()

Por outro lado, podemos deixar de fora a palavra-chave `function` e usar duas chaves logo após o nome da função:

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

É comum colocar funções em arquivos ou scripts. No entanto, elas também podem ser escritas diretamente no prompt do shell com cada comando em uma linha diferente—note PS2(>) indicando uma nova linha após uma quebra de linha:

```
$ greet() {
```

```
> greeting="Hello world!"
> echo $greeting
> }
```

Seja qual for o caso—e independentemente da sintaxe que escolhermos—, se decidirmos pular as quebras de linha e escrever uma função em apenas uma linha, os comandos devem ser separados por ponto e vírgula (observe o ponto e vírgula inclusive após o último comando):

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

O bash não reclamou quando pressionamos Enter, de modo que nossa função está pronta para ser chamada. Para invocar uma função, temos de digitar seu nome no terminal:

```
$ greet
Hello world!
```

Como no caso das variáveis e aliases, se quisermos que as funções sejam persistentes durante as reinicializações do sistema, temos de colocá-las em scripts de inicialização do shell, como `/etc/bash.bashrc` (global) ou `~/.bashrc` (local).

WARNING

Depois de adicionar aliases ou funções para qualquer arquivo de script de inicialização, é preciso executar `.` ou `source` nesses arquivos para que as alterações tenham efeito (caso você não queira fazer logout e login novamente ou reiniciar o sistema).

Variáveis integradas especiais do Bash

O *Bourne Again Shell* vem com um conjunto de variáveis especiais que são particularmente úteis para funções e scripts. Elas são especiais porque só podem ser referenciadas—e não atribuídas. Eis uma lista das mais relevantes:

`$?`

A referência desta variável se expande para o resultado do último comando executado. Um valor de `0` significa sucesso:

```
$ 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

Um valor diferente de 0 significa erro:

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

\$\$

Expande-se para o PID do shell (ID do processo):

```
$ 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
```

\$!

Expande-se para o PID do último trabalho em segundo plano:

```
$ 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 Lembre-se, o e comercial (&) é usado para iniciar processos em segundo plano.

Parâmetros posicionais \$0 a \$9

Expandem-se para os parâmetros ou argumentos que estão sendo passados para a função (alias ou script) — \$0 se expande para o nome do script ou shell.

Vamos criar uma função para demonstrar os parâmetros posicionais — note PS2 (>) indicando novas linhas após as quebras de linha:

```
$ special_vars() {  
> echo $0  
> echo $1  
> echo $2  
> echo $3  
}
```

Agora, vamos invocar a função (`special_vars`) passando três parâmetros para ela (`debian`, `ubuntu`, `zorin`):

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

Tudo funcionou como esperado.

Embora seja tecnicamente possível passar parâmetros posicionais para aliases, não é lá muito prático, já que — com aliases — os parâmetros posicionais são sempre passados no final:

WARNING

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

Outras variáveis integradas especiais do Bash incluem:

\$#

Expande-se para o número de argumentos passados para o comando.

\$@, \$*

Expandem-se para os argumentos passados para o comando.

\$_

Expande-se para o último parâmetro ou o nome do script (entre outras coisas; consulte `man bash` para saber mais!):

Variáveis em funções

Obviamente, as variáveis podem ser usadas dentro de funções.

Para provar, desta vez vamos criar um novo arquivo vazio chamado `funed` e colocar nele a seguinte função:

```
editors() {
    editor=emacs

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

Como você já deve ter adivinhado, precisamos primeiro originar o arquivo para poder invocar a função:

```
$ . funed
```

E agora podemos testá-la:

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

Como você pode perceber, para que a função `editors` funcione corretamente, a variável `editor` deve primeiro ser definida. O escopo dessa variável é local para o shell atual e podemos referenciá-la enquanto a sessão durar:

```
$ echo $editor
emacs
```

Juntamente com as variáveis locais, também podemos incluir variáveis de ambiente em nossa função:

```
editors() {
    editor=emacs

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

```
}
```

```
editors
```

Observe como desta vez decidimos chamar a função de dentro do próprio arquivo (`editors` na última linha). Dessa forma, quando o arquivo for aberto e executado, a função também será chamada — tudo de uma vez:

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

Parâmetros posicionais em funções

Ocorre algo semelhante com os parâmetros posicionais.

Podemos passá-los para funções de dentro do arquivo ou script (observe a última linha: `editors tortoise`):

```
editors() {

editor=emacs

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

editors tortoise
```

Fazemos source no arquivo para provar que funciona:

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

E também podemos passar parâmetros posicionais para funções na linha de comando. Para provar, removemos a última linha do arquivo:

```
editors() {

editor=emacs
```

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

Em seguida, temos de fazer source no arquivo:

```
$ . funed
```

Finalmente, invocamos a função com `tortoise` como o parâmetro posicional `$1` na linha de comando:

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

Funções em scripts

As funções são encontradas principalmente em scripts do Bash.

É fácil transformar nosso arquivo `funed` em um script (vamos chamá-lo de `funed.sh`):

```
#!/bin/bash

editors() {

editor=emacs

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

editors tortoise
```

É só isso! Adicionamos apenas duas linhas:

- A primeira linha é o *shebang* e define qual programa irá interpretar o script: `#!/bin/bash`. Curiosamente, esse programa é o próprio bash.
- A última linha é simplesmente a invocação da função.

Agora só resta uma coisa — tornar o script executável:

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

E agora ele está pronto para ser executado:

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

NOTE | Você aprenderá tudo sobre *criar scripts do shell* nas próximas lições.

Uma função dentro de um alias

Como dito acima, podemos colocar uma função dentro de um alias:

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

Este longo valor de alias merece uma explicação. Vamos decompô-lo:

- Primeiro, temos a função em si: `gr8_ed() { echo $1 is a great text editor; unset -f gr8_ed; }`
- O último comando da função — `unset -f gr8_ed` — remove a função definida, de forma que ela não permanece na sessão atual do `bash` depois que o alias é chamado.
- Por fim, para ter uma invocação de alias bem-sucedida, devemos primeiro invocar também a função: `gr8_ed`.

Vamos invocar o alias e provar que ele funciona:

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

Como mostrado em `unset -f gr8_ed` acima, o comando `unset` não é usado somente para remover as variáveis definidas, mas também as funções. Na verdade, existem opções específicas:

unset -v

para as variáveis

unset -f

para as funções

Se usado sem opções, o `unset` tenta primeiro remover uma variável e — se falhar — ele tenta remover uma função.

Uma função dentro de uma função

Agora, digamos que queremos comunicar duas coisas a `user2` toda vez que ele se logar no sistema:

- Dizer oi e recomendar/elogiar um editor de texto.
- Como ele tem posto muitos arquivos de vídeo Matroska na pasta `$HOME/Video` folder, também queremos adverti-la de que existe um limite.

Para cumprir esse objetivo, colocamos as duas funções a seguir em `/home/user2/.bashrc`:

A primeira função (`check_vids`) faz a verificação dos arquivos `.mkv` e emite o aviso:

```
check_vids() {
    ls -1 ~/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` faz três coisas:

- Lista os arquivos `mkv` em `~/Video` enviando a saída — e quaisquer erros — para o chamado *bitbucket* (`/dev/null`).
- Testa a saída do comando anterior.
- Dependendo do resultado do teste, ecoa uma mensagem (entre duas opções).

A segunda função é uma versão modificada da nossa função `editors`:

```
editors() {
    editor=emacs
```

```
echo "Hi, $USER!"  
echo "$editor is more than a text editor!"  
  
check_vids  
}  
  
editors
```

É importante observar duas coisas:

- O último comando de `editors` invoca `check_vids`, de modo que as duas funções são encadeadas: A saudação, a recomendação e a verificação e aviso são executados em sequência.
- `editors` é o ponto de entrada para a seqüência de funções, portanto é invocado na última linha (`editors`).

Agora, vamos entrar como `user2` para provar que funciona:

```
# 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.
```

Exercícios Guiados

1. Complete a tabela com “Sim” ou “Não” levando em conta as capacidades dos aliases e funções:

Recurso	Aliases?	Funções?
Variáveis locais podem ser usadas		
Variáveis de ambiente podem ser usadas		
Podem ser escapadas com \		
Podem ser recursivas		
Muito produtivas quando usadas com parâmetros posicionais		

2. Escreva o comando que lista todos os aliases do sistema:

3. Escreva um alias chamado `logg` para listar todos os arquivos `ogg` em `~/Music` — um por linha:

4. Invoque o alias para provar que funciona:

5. Agora, modifique o alias para que ele exiba o nome do usuário da sessão e dois pontos antes da listagem:

1. Invoque-o novamente para provar que esta nova versão também funciona:

2. Liste novamente todos os aliases e confira se o alias `logg` aparece na listagem:

3. Remova o alias:

4. Estude as colunas “Nome do Alias” e “Comando(s) com Alias” e atribua corretamente os aliases

a seus valores:

Nome do Alias	Comando(s) com Alias	Atribuição do Alias
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	

5. Como `root`, escreva uma função chamada `my_fun` em `/etc/bash.bashrc`. A função deve dizer olá ao usuário e informá-lo de qual é seu caminho. Invoque-a para que o usuário receba ambas as mensagens a cada vez que fizer login:

6. Faça login como `user2` para verificar se funciona:

7. Escreva a mesma função em apenas uma linha:

8. Invoque a função:

9. Remova a função:

10. Esta é uma versão modificada da função `special_vars`:

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

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

Este é o comando que usamos para invocá-la:

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

Adivinhe as saídas:

Referência	Valor
echo \$#	
echo \$_	
echo \$1	
echo \$4	
echo \$6	
echo \$7	
echo \$_	
echo \$@	
echo \$?	

11. Com base na função de exemplo (`check_vids`) da seção “Uma função dentro de uma função”, escreva uma função chamada `check_music` a ser incluída em um script de inicialização do bash que aceite parâmetros posicionais, sendo assim possível modificá-la facilmente:

- tipo de arquivo a ser verificado: `ogg`
- diretório em que os arquivos são salvos: `~/Music`
- tipo de arquivo sendo guardado: `music`
- número de arquivos salvos: `7`

Exercícios Exploratórios

1. Funções somente leitura são aquelas cujo conteúdo não podemos modificar. Faça uma pesquisa sobre *funções somente leitura* e preencha a seguinte tabela:

Nome da função	Torná-la somente leitura	Listar todas as funções somente leitura
my_fun		

2. Pesquise na web como modificar PS1, além de outras informações necessárias para escrever uma função chamada fyi (a ser posta em um script de inicialização) que forneça ao usuário as seguintes informações:

- nome do usuário
- diretório inicial
- nome do host
- tipo de sistema operacional
- caminho de pesquisa para executáveis
- diretório de email
- frequência de verificação do email
- quantas camadas de shell tem a sessão atual
- prompt (deve ser modificado para exibir <user>@<host-date>)

Resumo

Nesta lição, você aprendeu:

- Tanto aliases quanto funções são recursos importantes do shell que permitem encapsular blocos de código recorrentes.
- Os aliases são úteis para criar versões mais curtas de comandos longos e/ou complicados.
- As funções são procedimentos que implementam lógica e permitem automatizar tarefas, especialmente quando usadas em scripts.
- A sintaxe para criar aliases e funções.
- Como concatenar diversos comandos usando ponto e vírgula (;).
- Como usar aspas corretamente com os aliases.
- Como tornar persistentes os aliases e funções.
- Variáveis especiais nativas do Bash: \$?, \$\$, \$!, parâmetros posicionais (\$0-\$9), \$#,\$@, \${*} e \${_}.
- Como usar variáveis e parâmetros posicionais com funções.
- Como usar funções em scripts.
- Como invocar uma função a partir de um alias.
- Como invocar uma função a partir de outra função.
- Noções básicas para a criação de um script bash.

Comandos e palavras-chave usados nesta lição:

alias

Criar aliases.

unalias

Remover aliases.

cd

Mudar de diretório.

grep

Imprimir linhas de acordo com um padrão.

function

Palavra-chave do shell para criar funções.

.

Buscar e executar um arquivo.

source

Buscar e executar um arquivo.

ps

Exibir um instantâneo dos processos atuais.

echo

Exibir uma linha de texto.

chmod

Alterar os bits de modo de um arquivo, por exemplo para torná-lo executável.

unset

Remover variáveis e funções definidas.

su

Alterar o ID do usuário ou torná-lo superusuário.

Respostas aos Exercícios Guiados

1. Complete a tabela com “Sim” ou “Não” levando em conta as capacidades dos aliases e funções:

Recurso	Aliases?	Funções?
Variáveis locais podem ser usadas	Sim	Sim
Variáveis de ambiente podem ser usadas	Sim	Sim
Podem ser escapadas com \	Sim	Não
Podem ser recursivas	Sim	Sim
Muito produtivas quando usadas com parâmetros posicionais	Não	Sim

2. Escreva o comando que lista todos os aliases do sistema:

```
alias
```

3. Escreva um alias chamado logg para listar todos os arquivos ogg em ~/Music — um por linha:

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

4. Invoque o alias para provar que funciona:

```
logg
```

5. Agora, modifique o alias para que ele exiba o nome do usuário da sessão e dois pontos antes da listagem:

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

6. Invoque-o novamente para provar que esta nova versão também funciona:

```
logg
```

7. Liste novamente todos os aliases e confira se o alias `logg` aparece na listagem:

```
alias
```

8. Remova o alias:

```
unalias logg
```

9. Estude as colunas “Nome do Alias” e “Comando(s) com Alias” e atribua corretamente os aliases a seus valores:

Nome do Alias	Comando(s) com Alias	Atribuição do Alias
b	bash	<code>alias b=bash</code>
bash_info	<code>which bash + echo "\$BASH_VERSION"</code>	<code>alias bash_info='which bash; echo "\$BASH_VERSION"'</code>
kernel_info	<code>uname -r</code>	<code>alias kernel_info='uname -r'</code>
greet	<code>echo Hi, \$USER!</code>	<code>alias greet='echo Hi, \$USER'</code>
computer	<code>pc=slimbook + echo My computer is a \$pc</code>	<code>alias computer='pc=slimbook; echo My computer is a \$pc'</code>

NOTE As aspas simples podem ser substituídas por aspas duplas.

10. Como `root`, escreva uma função chamada `my_fun` em `/etc/bash.bashrc`. A função deve dizer olá ao usuário e informá-lo de qual é seu caminho. Invoque-a para que o usuário receba ambas as mensagens a cada vez que fizer login:

Opção A:

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

my_fun

Opção B:

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

11. Faça login como `user2` para verificar se funciona:

su - user2

12. Escreva a mesma função em apenas uma linha:

Opção A:

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

Opção B:

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

13. Invoque a função:

my_fun

14. Remova a função:

unset -f my_fun

15. Esta é uma versão modificada da função `special_vars`:

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

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

Este é o comando que usamos para invocá-la:

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

Adivinhe as saídas:

Referência	Valor
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. Com base na função de exemplo (`check_vids`) da seção “Uma função dentro de uma função”, escreva uma função chamada `check_music` a ser incluída em um script de inicialização do bash que aceite parâmetros posicionais, sendo assim possível modificá-la facilmente:

- tipo de arquivo a ser verificado: `ogg`
- diretório em que os arquivos são salvos: `~/Music`
- tipo de arquivo sendo guardado: `music`
- número de arquivos salvos: `7`

```
check_music() {  
    ls -1 ~/.$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
```

Respostas aos Exercícios Exploratórios

1. Funções somente leitura são aquelas cujo conteúdo não podemos modificar. Faça uma pesquisa sobre *funções somente leitura* e preencha a seguinte tabela:

Nome da função	Torná-la somente leitura	Listar todas as funções somente leitura
my_fun	readonly -f my_fun	readonly -f

2. Pesquise na web como modificar PS1, além de outras informações necessárias para escrever uma função chamada fyi (a ser posta em um script de inicialização) que forneça ao usuário as seguintes informações:

- nome do usuário
- diretório inicial
- nome do host
- tipo de sistema operacional
- caminho de pesquisa para executáveis
- diretório de email
- frequência de verificação do email
- quantas camadas de shell tem a sessão atual
- prompt (deve ser modificado para exibir <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

```



**Linux
Professional
Institute**

105.2 Editar e escrever scripts simples

Referência ao LPI objectivo

LPIC-1 5.0, Exam 102, Objective 105.2

Peso

4

Áreas chave de conhecimento

- Usar a sintaxe padrão sh (repetição, testes).
- Usar a substituição de comandos.
- Valores retornados por um sucesso ou falha de teste ou outra informação fornecida por um comando.
- Executar comandos encadeados.
- Enviar mensagens para o superusuário.
- Selecionar corretamente o interpretador de script através da linha shebang (#!).
- Gerenciar a localização, propriedade, permissão e permissão suid dos scripts.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

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

- ||
- &&



**Linux
Professional
Institute**

105.2 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	105 Shells e scripts do Shell
Objetivo:	105.2 Personalizar ou criar scripts simples
Lição:	1 de 2

Introdução

O ambiente de shell do Linux permite o uso de arquivos—chamados *scripts*—contendo comandos de qualquer programa disponível no sistema combinados a comandos internos do shell para automatizar as tarefas de um usuário e/ou do sistema. Com efeito, muitas das tarefas de manutenção do sistema operacional são executadas por scripts que consistem em sequências de comandos, estruturas de decisão e loops condicionais. Embora os scripts sejam na maioria das vezes destinados a tarefas relacionadas ao próprio sistema operacional, eles também são úteis para tarefas orientadas ao usuário, como renomeação em massa de arquivos, coleta e análise de dados ou qualquer outra atividade repetitiva de linha de comando.

Scripts nada mais são do que arquivos de texto que se comportam como programas. Um programa de verdade—the interpretador—lê e executa as instruções listadas no arquivo de script. O interpretador também pode iniciar uma sessão interativa na qual os comandos—including scripts—are lidos e executados à medida que são inseridos, como é o caso das sessões de shell do Linux. Os arquivos de script podem agrupar essas instruções e comandos quando se tornam complexos demais para serem implementados como um alias ou uma função de shell personalizada. Além disso, os arquivos de script podem ser mantidos como programas convencionais e, sendo apenas arquivos de texto, podem ser criados e modificados com qualquer

editor de texto simples.

Estrutura e execução de scripts

Basicamente, um arquivo de script é uma seqüência ordenada de comandos que devem ser executados por um interpretador de comandos. Um interpretador pode ler um arquivo de script de vários modos, e há maneiras distintas de fazer isso em uma sessão de shell Bash, mas o interpretador padrão para um arquivo de script será o indicado na primeira linha do script, logo após os caracteres `#!` (conhecidos como *shebang*). Em um script com instruções para o shell Bash, a primeira linha deve ser `#!/bin/bash`. Quando essa linha é indicada, o interpretador de todas as instruções do arquivo será `/bin/bash`. Tirando a primeira linha, todas as outras linhas que começam com o caractere hash `#` serão ignoradas e, assim, podem ser usadas para lembretes e comentários. As linhas em branco também são ignoradas. Um arquivo muito conciso de script do shell pode, portanto, ser escrito da seguinte maneira:

```
#!/bin/bash

# A very simple script

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

date +%H:%M
```

Este script tem apenas duas instruções para o interpretador `/bin/bash`: o comando interno `echo` e o comando `date`. A maneira mais básica de rodar um arquivo de script é executar o interpretador com o caminho do script como argumento. Assim, supondo que o exemplo anterior foi salvo em um arquivo de script chamado `script.sh` no diretório atual, ele será lido e interpretado pelo Bash com o seguinte comando:

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

O comando `echo` adiciona automaticamente uma nova linha após exibir o conteúdo, mas a opção `-n` suprime esse comportamento. Portanto, usar `echo -n` no script fará com que a saída de ambos os comandos apareça na mesma linha:

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

Embora não seja obrigatório, o sufixo `.sh` ajuda a identificar os scripts do shell ao listar e pesquisar arquivos.

TIP

O Bash chama qualquer comando indicado após o `#!` como interpretador do arquivo de script. Pode ser útil, por exemplo, empregar o shebang para outras linguagens de script, como `Python` (`#!/usr/bin/python`), `Perl` (`#!/usr/bin/perl`) ou `awk` (`#!/usr/bin/awk`).

Se o arquivo de script se destina a ser executado por outros usuários do sistema, é importante verificar se as permissões de leitura adequadas estão definidas. O comando `chmod o+r script.sh` concede permissão de leitura a todos os usuários do sistema, permitindo-lhes executar `script.sh` com o caminho para o arquivo de script como o argumento do comando `bash`. Como alternativa, o arquivo de script pode ter a permissão do bit de execução definida para que o arquivo possa ser executado como um comando convencional. O bit de execução é ativado no arquivo de script com o comando `chmod`:

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

Com o bit de execução habilitado, o arquivo de script chamado `script.sh` no diretório atual pode ser executado diretamente com o comando `./script.sh`. Os scripts colocados em um diretório listado na variável de ambiente `PATH` também estarão acessíveis sem seu caminho completo.

WARNING

Um script que executa ações restritas pode ter sua permissão SUID ativada e, portanto, os usuários comuns também podem executar o script com privilégios de root. Nesse caso, é muito importante garantir que nenhum usuário além do root tenha permissão para escrever no arquivo. Caso contrário, um usuário comum poderá modificar o arquivo para realizar operações arbitrárias e potencialmente prejudiciais.

O posicionamento e o recuo dos comandos em arquivos de script não são muito rígidos. Cada linha de um script de shell será executada como um comando de shell comum, na mesma sequência em que a linha aparece no arquivo de script, e as mesmas regras que se aplicam ao prompt do shell também se aplicam a cada linha de script individualmente. É possível colocar dois ou mais comandos na mesma linha, separados por ponto e vírgula:

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

Embora esse formato possa ser conveniente às vezes, seu uso é opcional, pois os comandos sequenciais podem ser postos em linhas separadas, sendo executados exatamente como se estivessem separados por ponto-e-vírgula. Em outras palavras, o ponto-e-vírgula pode ser

substituído por um caractere de nova linha nos arquivos de script do Bash.

Quando um script é executado, os comandos nele contidos não são executados diretamente na sessão atual, mas sim por um novo processo do Bash, chamado *sub-shell*. Isso evita que o script sobrescreva as variáveis de ambiente da sessão atual e faça modificações indesejadas nela. Se o objetivo é executar o conteúdo do script na sessão atual do shell, ele deve ser executado com `source script.sh` ou `. script.sh` (note que há um espaço entre o ponto e o nome do script).

Assim como acontece com a execução de qualquer outro comando, o prompt do shell só estará disponível novamente quando o script finalizar sua execução e seu código de status de saída estiver disponível na variável `$?`. Para mudar esse comportamento, de forma que o shell atual também se encerre quando o script for concluído, o script — ou qualquer outro comando — pode ser precedido pelo comando `exec`. Esse comando também substitui o código de status de saída da sessão do shell atual pelo seu próprio.

Variáveis

As variáveis nos scripts de shell se comportam da mesma maneira que nas sessões interativas, visto que o interpretador é o mesmo. Por exemplo, o formato `SOLUTION=42` (sem espaços ao redor do sinal de igual) atribuirá o valor `42` à variável de nome `SOLUTION`. Por convenção, letras maiúsculas são usadas para nomes de variáveis, mas não é obrigatório. Os nomes das variáveis não podem, entretanto, começar com caracteres não alfabéticos.

Além das variáveis comuns criadas pelo usuário, os scripts do Bash também possuem um conjunto de variáveis especiais chamadas *parâmetros*. Ao contrário das variáveis comuns, os nomes dos parâmetros começam com um caractere não alfabético que designa sua função. Os argumentos passados para um script e outras informações úteis são armazenados em parâmetros como `$0`, `$*`, `$?` etc., onde o caractere após o cifrão indica a informação a ser obtida:

\$*

Todos os argumentos passados para o script.

\$@

Todos os argumentos passados para o script. Se usado com aspas duplas, como em `"$@"`, todos os argumentos serão colocados entre aspas duplas.

\$#

O número de argumentos.

\$0

O nome do arquivo de script.

\$!

PID do último programa executado.

\$\$

PID do shell atual.

\$?

Código de status de saída numérico do último comando concluído. Para processos POSIX padrão, um valor numérico de `0` indica que o último comando foi executado com sucesso, o que também se aplica a scripts do shell.

Um *parâmetro posicional* é um parâmetro denotado por um ou mais dígitos diferentes do dígito único `0`. Por exemplo, a variável `$1` corresponde ao primeiro argumento dado ao script (parâmetro posicional um), `$2` corresponde ao segundo argumento e assim por diante. Se a posição de um parâmetro for maior que nove, ele deve ser referenciado com chaves, como em `${10}`, `${11}` etc.

As variáveis comuns, por outro lado, têm como função armazenar valores inseridos manualmente ou a saída gerada por outros comandos. O comando `read`, por exemplo, pode ser usado dentro do script para solicitar informações ao usuário durante a execução do script:

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

O valor retornado será armazenado na variável `ANSWER`. Se o nome da variável não for fornecido, o nome da variável `REPLY` será usado por padrão. Também é possível usar o comando `read` para ler mais de uma variável simultaneamente:

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

Neste caso, cada termo separado por espaços será atribuído às variáveis `NAME` e `SURNAME` respectivamente. Se o número de termos dados for maior que o número de variáveis, os termos excedentes serão armazenados na última variável. O próprio `read` pode exibir a mensagem para o usuário com a opção `-p`, tornando o comando `echo` redundante nesse caso:

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

Os scripts que executam tarefas do sistema geralmente requerem informações fornecidas por

outros programas. A *notação crase* (backtick) pode ser usada para armazenar a saída de um comando em uma variável:

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

No exemplo, a saída do comando `uname -o` será armazenada na variável `OS`. Um resultado idêntico será produzido com `$()`:

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

O comprimento de uma variável, ou seja, a quantidade de caracteres que ela contém, é retornado acrescentando-se um hash # antes do nome da variável. Esse recurso, no entanto, requer o uso da sintaxe das chaves para indicar a variável:

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

O Bash também apresenta variáveis de matriz (array) unidimensionais, de forma que um conjunto de elementos relacionados pode ser armazenado com um único nome de variável. Cada elemento de uma matriz possui um índice numérico, que deve ser usado para escrever e ler valores no elemento correspondente. Ao contrário das variáveis comuns, as matrizes devem ser declaradas com o comando interno do Bash `declare`. Por exemplo, para declarar uma variável chamada `SIZES` como uma matriz:

```
$ declare -a SIZES
```

As matrizes também podem ser declaradas implicitamente quando preenchidas a partir de uma lista predefinida de itens, usando a notação de parênteses:

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

No exemplo, os dois grandes valores inteiros foram armazenados na matriz `SIZES`. Os elementos da matriz devem ser referenciados usando chaves e colchetes, caso contrário o Bash não alterará nem exibirá o elemento corretamente. Como os índices da matriz começam em 0, o conteúdo do primeiro elemento está em `${SIZES[0]}` , o segundo em `${SIZES[1]}` e assim por diante:

```
$ echo ${SIZES[0]}
1048576
$ echo ${SIZES[1]}
1073741824
```

Diferente da leitura, a alteração do conteúdo de um elemento da matriz é realizada sem as chaves (por exemplo, `SIZES[0]=1048576`). Como no caso das variáveis comuns, o comprimento de um elemento em uma matriz é retornado com o caractere hash (por exemplo, `#$SIZES[0]` para o comprimento do primeiro elemento da matriz `SIZES`). O número total de elementos em uma matriz é retornado se `@` ou `*` forem usados como o índice:

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

As matrizes também podem ser declaradas usando-se, como elementos iniciais, a saída de um comando, por meio da substituição de comando. O exemplo a seguir mostra como criar uma matriz do Bash cujos elementos são os sistemas de arquivos suportados pelo sistema atual:

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

O comando `cut -f 2 < /proc/filesystems` exibe todos os sistemas de arquivos atualmente suportados pelo kernel em execução (listados na segunda coluna do arquivo `/proc/filesystems`), de forma que a matriz `FS` agora contém um elemento para cada sistema de arquivos suportado. Qualquer conteúdo de texto pode ser usado para inicializar uma matriz, já que, por padrão, quaisquer termos delimitados por caracteres de *espaço*, *tabulação* ou *nova linha* se tornarão um elemento de matriz.

TIP

O Bash trata cada caractere do `$IFS` (*Input Field Separator* ou separador de campos) de uma variável de ambiente como um delimitador. Para alterar o delimitador de campo apenas para caracteres de nova linha, por exemplo, a variável `IFS` deve ser redefinida com o comando `IFS=$'\n'`.

Expressões aritméticas

O Bash oferece um método prático para realizar operações aritméticas de números inteiros com o comando interno `expr`. Duas variáveis numéricas, `$VAL1` e `$VAL2`, por exemplo, podem ser adicionadas junto com o seguinte comando:

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

O valor resultante do exemplo estará disponível na variável \$SUM. O comando expr pode ser substituído por \$(()), de forma que o exemplo anterior pode ser reescrito como SUM=\$((\$VAL1 + \$VAL2)). Expressões com potenciação são igualmente permitidas com o operador de duplo asterisco, de forma que a declaração de matriz anterior SIZES=(1048576 1073741824) poderia ser reescrita como SIZES=(\$((1024**2)) \$((1024**3))).

A substituição de comandos também pode ser usada em expressões aritméticas. Por exemplo, o arquivo /proc/meminfo contém informações detalhadas sobre a memória do sistema, incluindo o número de bytes livres na RAM:

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

O exemplo mostra como o comando sed pode ser usado para analisar o conteúdo de /proc/meminfo dentro da expressão aritmética. A segunda linha do arquivo /proc/meminfo contém a quantidade de memória livre em milhares de bytes, então a expressão aritmética multiplica esse valor por 1000 para obter o número de bytes livres na RAM.

Execução condicional

Alguns scripts geralmente não se destinam a executar todos os comandos no arquivo de script, mas apenas aqueles que correspondem a critérios predefinidos. Por exemplo, um script de manutenção pode enviar uma mensagem de aviso ao email do administrador somente se a execução de um comando falhar. O Bash fornece métodos específicos para avaliar o sucesso da execução de comandos, além de estruturas condicionais gerais, mais semelhantes às encontradas nas linguagens de programação populares.

Ao separar os comandos com &&, o comando à direita será executado apenas se o comando à esquerda não encontrar um erro, ou seja, se seu status de saída for igual a 0:

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

O comportamento oposto ocorre se os comandos estiverem separados por ||. Nesse caso, o comando a seguir será executado apenas se o comando anterior encontrar um erro, ou seja, se seu código de status de retorno for diferente de 0.

Um dos recursos mais importantes de todas as linguagens de programação é a capacidade de executar comandos dependendo de condições previamente definidas. A maneira mais direta de

executar comandos condicionalmente é usar o comando interno do Bash `if`, que executa um ou mais comandos somente se o comando fornecido como argumento retornar um código de status 0 (sucesso). Outro comando, `test`, pode ser usado para avaliar diversos critérios especiais diferentes, sendo assim usado principalmente em conjunto com `if`. No exemplo a seguir, a mensagem `Confirmed: /bin/bash is executable.` será exibida se o arquivo `/bin/bash` existir e for executável:

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

A opção `-x` faz com que o comando `test` retorne um código de status 0 apenas se o caminho fornecido for um arquivo executável. O exemplo a seguir mostra outra maneira de obter exatamente o mesmo resultado, já que os colchetes podem ser usados como substitutos para `test`:

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

A instrução `else` é opcional para a estrutura `if` e pode, se presente, definir um comando ou sequência de comandos a serem executados se a expressão condicional não for verdadeira:

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

As estruturas `if` sempre devem terminar com `fi`, de forma que o interpretador Bash saiba onde os comandos condicionais terminam.

Saída do script

Mesmo quando a finalidade de um script envolve apenas operações orientadas a arquivos, é importante exibir mensagens relacionadas ao progresso na saída padrão, para que o usuário seja informado sobre quaisquer problemas e possa, eventualmente, usar essas mensagens para gerar logs de operação.

O comando interno do Bash `echo` é comumente usado para exibir strings de texto simples, mas ele

também oferece alguns recursos estendidos. Com a opção `-e`, o comando `echo` é capaz de exibir caracteres especiais usando sequências de escape (uma sequência de barra invertida designando um caractere especial). Por exemplo:

```
#!/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"
```

Embora o uso de aspas seja opcional ao se usar `echo` sem opções, é necessário adicioná-las ao incluir a opção `-e`; caso contrário, os caracteres especiais podem não ser lidos corretamente. No script anterior, ambos os comandos `echo` usam o caractere de tabulação `\t` para alinhar o texto, resultando na seguinte saída:

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

O caractere de nova linha `\n` pode ser usado para separar as linhas da saída, de forma que exatamente a mesma saída é obtida combinando-se os dois comandos `echo` em um só:

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

Embora adequado para exibir a maioria das mensagens de texto, o comando `echo` pode não ser o melhor para padrões de texto mais específicos. O comando interno do Bash `printf` oferece mais controle sobre a exibição das variáveis. O comando `printf` usa o primeiro argumento como formato da saída, onde os marcadores serão substituídos pelos argumentos seguintes na ordem em que aparecem na linha de comando. Assim, a mensagem do exemplo anterior poderia ser gerada com o seguinte comando `printf`:

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

O espaço reservado `%s` destina-se ao conteúdo de texto (será substituído pela variável `$OS`) e o espaço reservado `%d` destina-se a números inteiros (será substituído pelo número resultante de

megabytes livres na RAM). O `printf` não acrescenta um caractere de nova linha no final do texto, então o caractere de nova linha `\n` deve ser posto ao fim do padrão, se necessário. Todo o padrão deve ser interpretado como um único argumento e, portanto, deve ser posto entre aspas.

TIP

O formato de substituição do espaço reservado realizada por `printf` pode ser personalizado com o mesmo formato usado pela função `printf` da linguagem de programação C. A referência completa para a função `printf` pode ser encontrada em sua página de manual, acessada com o comando `man 3 printf`.

Com `printf`, as variáveis são postas fora do padrão de texto, o que torna possível armazenar o padrão de texto em uma variável separada:

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

Este método é particularmente útil para exibir formatos de saída distintos, dependendo dos requisitos do usuário. Fica mais fácil, por exemplo, produzir um script que use um padrão de texto distinto se o usuário precisar de uma lista CSV (valores separados por vírgula) em vez de uma mensagem de saída padrão.

Exercícios Guiados

1. A opção `-s` para o comando `read` é útil para inserir senhas, pois não mostra o conteúdo que está sendo digitado na tela. Como o comando `read` pode ser usado para armazenar os dados inseridos pelo usuário na variável `PASSWORD` enquanto oculta o conteúdo digitado?

2. A única finalidade do comando `whoami` é exibir o nome do usuário que o chamou, de modo que ele é usado principalmente dentro de scripts para identificar o usuário que o está executando. Dentro de um script Bash, como a saída do comando `whoami` pode ser armazenada na variável chamada `WHO`?

3. Qual operador do Bash deve estar entre os comandos `apt-get dist-upgrade` e `systemctl reboot` se o usuário `root` quiser executar `systemctl reboot` apenas se `apt-get dist-upgrade` for concluído com sucesso?

Exercícios Exploratórios

- Depois de tentar executar um script Bash recém-criado, um usuário recebe a seguinte mensagem de erro:

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

Considerando que o arquivo `./script.sh` foi criado pelo mesmo usuário, qual seria a provável causa desse erro?

- Suponha que um arquivo de script chamado `do.sh` seja executável e o link simbólico `undo.sh` aponte para ele. De dentro do script, como seria possível identificar se o nome do arquivo de chamada era `do.sh` ou `undo.sh`?

- Em um sistema com um serviço de email configurado corretamente, o comando `mail -s "Maintenance Error" root <<<"Scheduled task error"` envia o email de aviso ao usuário `root`. Esse comando pode ser usado em tarefas autônomas, como `cronjobs`, para informar o administrador do sistema sobre um problema inesperado. Escreva uma instrução com `if` para executar o comando `mail` acima mencionado somente se o status de saída do comando anterior — seja lá qual for — não for bem-sucedido.

Resumo

Esta lição cobre os conceitos básicos para compreender e escrever scripts de shell Bash. Os scripts de shell são uma parte essencial de qualquer distribuição Linux, pois oferecem uma maneira muito flexível de automatizar as tarefas do usuário e do sistema executadas no ambiente do shell. A lição trata dos seguintes tópicos:

- Estrutura dos scripts de shell e permissões corretas dos arquivos de script
- Parâmetros de script
- Uso de variáveis para ler as informações fornecidas pelo usuário e para armazenar a saída dos comandos
- Matrizes do Bash
- Testes simples e execução condicional
- Formatação da saída

Os comandos e procedimentos abordados foram:

- Notação interna do Bash para substituição de comandos, expansão de matrizes e expressões aritméticas
- Execução condicional de comandos com os operadores `||` e `&&`
- `echo`
- `chmod`
- `exec`
- `read`
- `declare`
- `test`
- `if`
- `printf`

Respostas aos Exercícios Guiados

1. A opção `-s` para o comando `read` é útil para inserir senhas, pois não mostra o conteúdo que está sendo digitado na tela. Como o comando `read` pode ser usado para armazenar os dados inseridos pelo usuário na variável `PASSWORD` enquanto oculta o conteúdo digitado?

```
read -s PASSWORD
```

2. A única finalidade do comando `whoami` é exibir o nome do usuário que o chamou, de modo que ele é usado principalmente dentro de scripts para identificar o usuário que o está executando. Dentro de um script Bash, como a saída do comando `whoami` pode ser armazenada na variável chamada `WHO`?

```
WHO=`whoami` ou WHO=$(whoami)
```

3. Qual operador do Bash deve estar entre os comandos `apt-get dist-upgrade` e `systemctl reboot` se o usuário root quiser executar `systemctl reboot` apenas se `apt-get dist-upgrade` for concluído com sucesso?

O operador `&&`, como em `apt-get dist-upgrade && systemctl reboot`.

Respostas aos Exercícios Exploratórios

1. Depois de tentar executar um script Bash recém-criado, um usuário recebe a seguinte mensagem de erro:

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

Considerando que o arquivo `./script.sh` foi criado pelo mesmo usuário, qual seria a provável causa desse erro?

O arquivo `./script.sh` não está com a permissão de execução habilitada.

2. Suponha que um arquivo de script chamado `do.sh` seja executável e o link simbólico `undo.sh` aponte para ele. De dentro do script, como seria possível identificar se o nome do arquivo de chamada era `do.sh` ou `undo.sh`?

A variável especial `$0` contém o nome de arquivo usado para chamar o script.

3. Em um sistema com um serviço de email configurado corretamente, o comando `mail -s "Maintenance Error" root <<<"Scheduled task error"` envia o email de aviso ao usuário root. Esse comando pode ser usado em tarefas autônomas, como `cronjobs`, para informar o administrador do sistema sobre um problema inesperado. Escreva uma instrução com `if` para executar o comando `mail` acima mencionado somente se o status de saída do comando anterior — seja lá qual for — não for bem-sucedido.

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



**Linux
Professional
Institute**

105.2 Lição 2

Certificação:	LPIC-1
Versão:	5.0
Tópico:	105 Shells e scripts do Shell
Objetivo:	105.2 Personalizar ou criar scripts simples
Lição:	2 de 2

Introdução

Os scripts de shell destinam-se geralmente a automatizar operações relacionadas a arquivos e diretórios — as mesmas operações que podem ser executadas manualmente na linha de comando. Porém, o alcance dos scripts de shell não se restringe apenas aos documentos do usuário, já que a configuração e interação com muitos aspectos de um sistema operacional Linux também são realizadas por meio de arquivos de script.

O shell Bash oferece muitos comandos internos úteis para criar scripts de shell, mas para aproveitar totalmente o poder desses scripts temos de combinar os comandos internos do Bash com os diversos utilitários de linha de comando disponíveis em um sistema Linux.

Testes ampliados

O Bash, como linguagem de script, é orientado sobretudo a arquivos, de forma que o comando interno do Bash `test` tem muitas opções para avaliar as propriedades dos objetos do sistema de arquivos (essencialmente arquivos e diretórios). Os testes que se concentram em arquivos e diretórios são úteis, por exemplo, para verificar se os arquivos e diretórios necessários para executar uma tarefa específica estão presentes e podem ser lidos. A seguir, associado a uma

construção condicional `if`, o conjunto apropriado de ações é executado caso o teste seja bem-sucedido.

O comando `test` avalia as expressões usando duas sintaxes diferentes: as expressões de teste podem ser dadas como um argumento para o comando `test` ou podem ser postas entre colchetes, caso em que o comando `test` é dado implicitamente. Assim, o teste para avaliar se `/etc` é um diretório válido pode ser escrito como `test -d /etc` ou como `[-d /etc]`:

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

Como confirmam os códigos de status de saída na variável especial `$?` — um valor de 0 significa que o teste foi bem-sucedido — ambas as formas avaliaram `/etc` como um diretório válido. Supondo-se que o caminho para um arquivo ou diretório foi armazenado na variável `$VAR`, as seguintes expressões podem ser usadas como argumentos para `test` ou entre colchetes:

-a "\$VAR"

Avalia se o caminho em `VAR` existe no sistema de arquivos e é um arquivo.

-b "\$VAR"

Avalia se o caminho em `VAR` é um arquivo de bloco especial.

-c "\$VAR"

Avalia se o caminho em `VAR` é um arquivo de caractere especial.

-d "\$VAR"

Avalia se o caminho em `VAR` é um diretório.

-e "\$VAR"

Avalia se o caminho em `VAR` existe no sistema de arquivos.

-f "\$VAR"

Avalia se o caminho em `VAR` existe e é um arquivo regular.

-g "\$VAR"

Avalia se o caminho em `VAR` tem permissão SGID.

-h "\$VAR"

Avalia se o caminho em VAR é um link simbólico.

-L "\$VAR"

Avalia se o caminho em VAR é um link simbólico (como -h).

-k "\$VAR"

Avalia se o caminho em VAR tem a permissão *sticky bit*.

-p "\$VAR"

Avalia se o caminho em VAR é um arquivo *pipe*.

-r "\$VAR"

Avalia se o caminho em VAR é legível pelo usuário atual.

-s "\$VAR"

Avalia se o caminho em VAR existe e não está vazio.

-S "\$VAR"

Avalia se o caminho em VAR é um arquivo de socket.

-t "\$VAR"

Avalia se o caminho em VAR está aberto em um terminal.

-u "\$VAR"

Avalia se o caminho em VAR tem permissão SUID.

-w "\$VAR"

Avalia se o caminho em VAR é gravável pelo usuário atual.

-x "\$VAR"

Avalia se o caminho em VAR é executável pelo usuário atual.

-o "\$VAR"

Avalia se o caminho em VAR é de propriedade do usuário atual.

-G "\$VAR"

Avalia se o caminho em VAR pertence ao grupo efetivo do usuário atual.

-N "\$VAR"

Avalia se o caminho em VAR foi modificado desde o último acesso.

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

Avalia se o caminho em VAR1 é mais recente que o caminho em VAR2, de acordo com as datas de modificação respectivas.

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

Avalia se o caminho em VAR1 é mais antigo que VAR2.

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

Esta expressão avalia como True (Verdadeiro) se o caminho em VAR1 é um link físico para VAR2.

Recomenda-se usar aspas duplas em torno de uma variável testada porque, se a variável por acaso estiver vazia, isso pode causar um erro de sintaxe para o comando `test`. As opções de teste requerem um argumento operando, e uma variável vazia sem aspas causaria um erro devido à falta de um argumento obrigatório. Também existem testes para variáveis de texto arbitrárias, descritos a seguir:

-z "\$TXT"

Avalia se a variável TXT está vazia (tamanho zero).

-n "\$TXT" or test "\$TXT"

Avalia se a variável TXT não está vazia.

"\$TXT1" = "\$TXT2" or "\$TXT1" == "\$TXT2"

Avalia se TXT1 e TXT2 são iguais.

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

Avalia se TXT1 e TXT2 não são iguais.

"\$TXT1" < "\$TXT2"

Avalia se TXT1 vem antes de TXT2, em ordem alfabética.

"\$TXT1" > "\$TXT2"

Avalia se TXT1 vem depois de TXT2, em ordem alfabética.

Linguagens diferentes podem ter regras diferentes para a ordenação alfabética. Para obter resultados consistentes, independentemente das configurações de localização do sistema no qual o script está sendo executado, é recomendável definir a variável de ambiente LANG como C, como

em `LANG=C`, antes de fazer operações que envolvam ordem alfabética. Essa definição também manterá as mensagens do sistema no idioma original e, portanto, deve ser usada apenas no escopo do script.

As comparações numéricas têm seu próprio conjunto de opções de teste:

\$NUM1 -lt \$NUM2

Avalia se `NUM1` é menor que `NUM2`.

\$NUM1 -gt \$NUM2

Avalia se `NUM1` é maior que `NUM2`.

\$NUM1 -le \$NUM2

Avalia se `NUM1` é menor ou igual a `NUM2`.

\$NUM1 -ge \$NUM2

Avalia se `NUM1` é maior ou igual a `NUM2`.

\$NUM1 -eq \$NUM2

Avalia se `NUM1` é igual a `NUM2`.

\$NUM1 -ne \$NUM2

Avalia se `NUM1` não é igual a `NUM2`.

Todos os testes podem receber os seguintes modificadores:

! EXPR

Avalia se a expressão `EXPR` é falsa.

EXPR1 -a EXPR2

Avalia se tanto `EXPR1` quanto `EXPR2` são verdadeiras.

EXPR1 -o EXPR2

Avalia se ao menos uma das duas expressões é verdadeira.

Outra construção condicional, `case`, pode ser vista como uma variação da construção `if`. A instrução `case` executa uma lista de comandos dados se um item especificado — por exemplo, o conteúdo de uma variável — puder ser encontrado em uma lista de itens separados por *pipes* (a barra vertical `|`) e encerrado por `)`. O exemplo de script a seguir mostra como a construção `case` pode ser usada para indicar o formato de pacote de software correspondente para uma determinada distribuição Linux:

```
#!/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."
```

Cada lista de padrões e comandos associados deve terminar com ;;, ;&, ou ;;&. O último padrão, um asterisco, será usado se não for encontrada uma correspondência para nenhum outro padrão anterior. A instrução `esac` (case de trás pra frente) conclui a construção `case`. Supondo que o script de amostra anterior se chame `script.sh` e seja executado com `opensuse` como primeiro argumento, a seguinte saída será gerada:

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

TIP

O Bash tem uma opção chamada `nocasematch` que ativa a correspondência de padrões sem distinção entre maiúsculas e minúsculas para a construção `case` e outros comandos condicionais. O comando interno `shopt` alterna os valores das configurações que controlam comportamentos opcionais do shell: `shopt -s` habilita (`set`) a opção fornecida e `shopt -u` desabilita (`unset`) a opção fornecida. Portanto, colocar `shopt -s nocasematch` antes da construção `case` permite encontrar padrões sem diferenciar maiúsculas de minúsculas. As opções modificadas por `shopt` afetarão apenas a sessão atual, de forma que as opções modificadas dentro de scripts em execução em um sub-shell — o que é a maneira padrão de executar um script — não afetarão as opções da sessão pai.

O item pesquisado e os padrões sofrem expansão de til, expansão de parâmetro, substituição de comando e expansão aritmética. Se o item pesquisado for especificado com aspas, elas serão

removidas antes do script tentar encontrar uma correspondência.

Construções de loop

Os scripts são freqüentemente usados como ferramenta para automatizar tarefas repetitivas, executando o mesmo conjunto de comandos até que seja verificado um critério de interrupção. O Bash tem três instruções de loop—`for`, `until` e `while`—projetadas para construções de loop ligeiramente distintas.

A construção `for` percorre uma lista dada de itens—geralmente uma lista de palavras ou quaisquer outros segmentos de texto separados por espaços—executando o mesmo conjunto de comandos em cada um desses itens. Antes de cada iteração, a instrução `for` atribui o item atual a uma variável, que pode então ser usada pelos comandos incluídos. O processo é repetido até que não restem mais itens. A sintaxe da construção `for` é:

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

`VARNAME` é um nome de variável arbitrária do shell e `LIST` é qualquer sequência de termos separados. Os caracteres delimitadores válidos que dividem os itens na lista são definidos pela variável de ambiente `IFS`, que são os caracteres *espaço*, *tabulação* e *nova linha* por padrão. A lista de comandos a serem executados é delimitada pelas instruções `do` e `done`, de modo que os comandos podem ocupar tantas linhas quantas forem necessárias.

No exemplo a seguir, o comando `for` pega cada item da lista fornecida—uma sequência de números—and os atribui à variável `NUM`, um de cada vez:

```
#!/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

No exemplo, uma construção aninhada `if` é usada em conjunto com uma expressão aritmética para avaliar se o número na variável `NUM` atual é par ou ímpar. Supondo-se que o script do exemplo anterior se chama `script.sh` e está no diretório atual, a seguinte saída será gerada:

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

O Bash também suporta um formato alternativo para construções `for`, com a notação de parênteses duplos. Essa notação se assemelha à sintaxe da instrução `for` da linguagem de programação C e é particularmente útil para trabalhar com matrizes:

```
#!/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
```

Este script gera exatamente a mesma saída do exemplo anterior. No entanto, em vez de usar a variável `NUM` para armazenar um item por vez, a variável `IDX` é empregada para rastrear o índice da matriz atual em ordem crescente, começando de 0 e continuando a adicionar enquanto esse número permanecer abaixo do número de itens na matriz `SEQ`. O item em si é recuperado de sua posição na matriz com `${SEQ[$IDX]}` .

Da mesma forma, a construção `until` executa uma sequência de comandos até que um comando

de teste — como o próprio comando `test` — seja encerrado com o status 0 (sucesso). Por exemplo, a mesma estrutura de loop do exemplo anterior pode ser implementada com `until` da seguinte forma:

```
#!/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
```

As construções `until` podem exigir mais instruções do que as construções `for`, mas podem ser mais adequadas para critérios de parada não-numéricos fornecidos pelas expressões `test` ou qualquer outro comando. É importante incluir ações que garantam um critério de parada válido, como o incremento de uma variável de contador, caso contrário o loop pode acabar sendo executado indefinidamente.

A instrução `while` é semelhante à instrução `until`, mas `while` continua repetindo o conjunto de comandos se o comando de teste terminar com o status 0 (sucesso). Portanto, a instrução `until [$IDX -eq ${#SEQ[*]}]` do exemplo anterior é equivalente a `while [$IDX -lt ${#SEQ[*]}]`, já que o loop deve ser repetido enquanto o índice da matriz for *menor que* o total de itens na matriz.

Um exemplo mais elaborado

Imagine que um usuário deseja sincronizar periodicamente uma coleção de seus arquivos e diretórios com outro dispositivo de armazenamento, montado em um ponto de montagem arbitrário no sistema de arquivos, e considera-se que um sistema de backup completo seria um exagero. Como esta é uma atividade que deve ser realizada periodicamente, trata-se de uma aplicação que vale a pena automatizar com um script de shell.

A tarefa é simples: sincronizar todos os arquivos e diretórios contidos em uma lista, de um diretório de origem informado como primeiro argumento do script para um diretório de destino informado como segundo argumento do script. Para ser mais fácil adicionar ou remover itens da lista, ela será mantida em um arquivo separado, `~/sync.list`, um item por linha:

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

O arquivo contém uma mistura de arquivos e diretórios, alguns deles com espaços em branco em seus nomes. Este é um cenário adequado para o comando interno do Bash `mapfile`, que analisa qualquer conteúdo de texto dado e cria uma variável de matriz a partir dele, colocando cada linha como um item de matriz individual. O arquivo do script será denominado `sync.sh` e conterá o seguinte script:

```
#!/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
```

A primeira ação realizada pelo script é redefinir dois parâmetros do shell com o comando `set`: a opção `-e` sai da execução imediatamente se um comando resultar em um status diferente de zero e a opção `-f` desabilita o globbing do nome do arquivo. Ambas as opções podem ser encurtadas como `-ef`. Esta não é uma etapa obrigatória, mas ajuda a diminuir a probabilidade de um comportamento inesperado.

As instruções reais do arquivo de script orientadas para a aplicação podem ser divididas em três partes:

1. Coletar e verificar os parâmetros do script

A variável `FILE` é o caminho para o arquivo que contém a lista de itens a serem copiados: `~/sync.list`. As variáveis `FROM` e `TO` são os caminhos de origem e destino, respectivamente. Como esses dois últimos parâmetros são fornecidos pelo usuário, eles passam por um teste de validação simples realizado pela construção `if`: se algum dos dois não for um diretório válido — determinado pelo teste `[! -d "$FROM" -o ! -d "$TO"]` — o script mostrará uma breve mensagem de ajuda e se encerrará com um status de saída 1.

2. Carregar lista de arquivos e diretórios

Após todos os parâmetros serem definidos, uma matriz contendo a lista de itens a serem copiados é criada com o comando `mapfile -t LIST < $FILE`. A opção `-t` do `mapfile` remove o caractere final de nova linha de cada linha antes de incluí-lo na variável de matriz chamada `LIST`. O conteúdo do arquivo indicado pela variável `FILE` — `~/sync.list` — é lido via redirecionamento de entrada.

3. Realizar a cópia e informar o usuário

Um loop `for` usando a notação de parênteses duplo percorre a matriz de itens, com a variável `IDX` controlando o incremento do índice. O comando `echo` informa o usuário sobre cada item que está sendo copiado. O caractere Unicode de escape — `\u2192` — para o caractere *seta direita* está presente na mensagem de saída, por isso opção `-e` do comando `echo` deve ser

usada. O comando `rsync` copia seletivamente apenas as partes modificadas do arquivo de origem, portanto seu uso é recomendado para esse tipo de tarefa. As opções `-q` e `-a` do `rsync`, condensadas em `-qa`, inibem as mensagens do `rsync` e ativam o modo *arquivar*, no qual todas as propriedades do arquivo são preservadas. A opção `--delete` faz com que o `rsync` exclua um item do destino que não exista mais na origem e, portanto, deve ser usada com cuidado.

Supondo-se que todos os itens da lista existem no diretório inicial da usuária `carol`, `/home/carol`, e que o diretório de destino `/media/carol/backup` aponta para um dispositivo de armazenamento externo montado, o comando `sync.sh /home/carol /media/carol/backup` gera a seguinte saída:

```
$ 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
```

O exemplo também supõe que o script é executado pelo root ou pela usuária `carol`, já que a maioria dos arquivos seria ilegível por outros usuários. Se `script.sh` não estiver dentro de um diretório listado na variável de ambiente PATH, ele deve ser especificado com seu caminho completo.

Exercícios Guiados

1. Como o comando `test` pode ser usado para verificar se o caminho do arquivo armazenado na variável `FROM` é mais recente do que um arquivo cujo caminho está armazenado na variável `TO`?

2. O script a seguir deveria imprimir uma sequência numérica de 0 a 9 mas, em vez disso, imprime 0 eternamente. O que deve ser feito para se obter a saída esperada?

```
#!/bin/bash

COUNTER=0

while [ $COUNTER -lt 10 ]
do
    echo $COUNTER
done
```

3. Suponha que um usuário escreveu um script que requer uma lista ordenada de nomes de usuário. A lista resultante é apresentada da seguinte forma em seu computador:

```
carol
Dave
emma
Frank
Grace
henry
```

No entanto, a mesma lista é ordenada assim no computador de seu colega:

```
Dave
Frank
Grace
carol
emma
henry
```

O que poderia explicar as diferenças entre as duas listas ordenadas?



Exercícios Exploratórios

1. Como todos os argumentos de linha de comando do script podem ser usados para inicializar uma matriz Bash?

2. Porque é que, contraintuitivamente, o comando `test 1 > 2` é avaliado como verdadeiro?

3. Como um usuário poderia alterar temporariamente o separador de campo padrão apenas para o caractere de nova linha, sem deixar de ser capaz de revertê-lo ao conteúdo original?

Resumo

Esta lição discorre mais profundamente sobre os testes disponíveis para o comando `test` e outras construções condicionais e de loop, necessárias para escrever scripts de shell mais elaborados. Um script de sincronização de arquivo simples é fornecido como exemplo de aplicação prática para um script de shell. A lição trata dos seguintes tópicos:

- Testes estendidos para as construções condicionais `if` e `case`.
- Construções de loop do shell: `for`, `until` e `while`.
- Iterações por meio de matrizes e parâmetros.

Os comandos e procedimentos abordados foram:

test

Realiza uma comparação entre os itens fornecidos ao comando.

if

Uma construção lógica usada em scripts para avaliar algo como verdadeiro ou falso e em seguida lançar a execução do comando com base nos resultados.

case

Avalia diversos valores em relação a uma única variável. A execução do comando do script é então realizada dependendo do resultado do comando `case`.

for

Repete a execução de um comando com base em um determinado critério.

until

Repete a execução de um comando até que uma expressão seja avaliada como falsa.

while

Repete a execução de um comando enquanto uma dada expressão for avaliada como verdadeira.

Respostas aos Exercícios Guiados

1. Como o comando `test` pode ser usado para verificar se o caminho do arquivo armazenado na variável `FROM` é mais recente do que um arquivo cujo caminho está armazenado na variável `TO`?

O comando `test "$FROM" -nt "$TO"` retorna um código de status 0 se o arquivo na variável `FROM` for mais recente que o arquivo na variável `TO`.

2. O script a seguir deveria imprimir uma sequência numérica de 0 a 9 mas, em vez disso, imprime 0 eternamente. O que deve ser feito para se obter a saída esperada?

```
#!/bin/bash

COUNTER=0

while [ $COUNTER -lt 10 ]
do
    echo $COUNTER
done
```

A variável `COUNTER` deve ser incrementada, o que pode ser feito com a expressão aritmética `COUNTER=$(($COUNTER + 1))`, até atingir os critérios de parada e encerrar o loop.

3. Suponha que um usuário escreveu um script que requer uma lista ordenada de nomes de usuário. A lista resultante é apresentada da seguinte forma em seu computador:

```
carol
Dave
emma
Frank
Grace
henry
```

No entanto, a mesma lista é ordenada assim no computador de seu colega:

```
Dave
Frank
Grace
carol
emma
```

henry

O que poderia explicar as diferenças entre as duas listas ordenadas?

A classificação baseia-se na localidade (idioma) do sistema atual. Para evitar inconsistências, as tarefas de classificação devem ser realizadas com a variável de ambiente `LANG` definida como `C`.

Respostas aos Exercícios Exploratórios

1. Como todos os argumentos de linha de comando do script podem ser usados para inicializar uma matriz Bash?

Os comandos `PARAMS=($*)` ou `PARAMS=("$@")` criam uma matriz chamada `PARAMS` com todos os argumentos.

2. Porque é que, contraintuitivamente, o comando `test 1 > 2` é avaliado como verdadeiro?

O operador `>` deve ser usado com testes de string, e não testes numéricos.

3. Como um usuário poderia alterar temporariamente o separador de campo padrão apenas para o caractere de nova linha, sem deixar de ser capaz de revertê-lo ao conteúdo original?

Uma cópia da variável `IFS` pode ser armazenada em outra variável: `OLDIFS=$IFS`. Em seguida o novo separador de linhas é definido com `IFS=$'\\n'` e a nova variável `IFS` pode ser revertida com `IFS=$OLDIFS`.



Tópico 106: Interfaces de usuário e Desktops



**Linux
Professional
Institute**

106.1 Instalar e configurar o X11

Referência ao LPI objectivo

[LPIC-1 version 5.0, Exam 102, Objective 106.1](#)

Peso

2

Áreas chave de conhecimento

- Entendimento da arquitetura do X11.
- Entendimento e conhecimento básico do arquivo de configuração do X Window.
- Substituir aspectos específicos da configuração do Xorg, como o layout de teclado.
- Entendimento dos componentes de um ambiente de desktop, como gerenciadores de display e gerenciadores de janelas.
- Controlar o acesso ao servidor X e exibir aplicativos em servidores X remotos.
- Noções do Wayland.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- /etc/X11/xorg.conf
- /etc/X11/xorg.conf.d/
- ~/.xsession-errors
- xhost
- xauth
- DISPLAY
- X



**Linux
Professional
Institute**

106.1 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	106 Interfaces de usuário e áreas de trabalho
Objetivo:	106.1 Instalar e configurar o X11
Lição:	1 de 1

Introdução

O X Window System é uma pilha de software usada para exibir texto e gráficos em uma tela. A aparência geral e o design de um cliente X não são ditados pelo X Window System, mas sim gerenciados por cada cliente X individual, um *gerenciador de janela* (por exemplo, Window Maker, Tab Window Manager) ou um *ambiente de desktop* completo como KDE, GNOME ou Xfce. Os ambientes de desktop serão abordados em uma lição posterior. Esta lição se concentrará na arquitetura subjacente e nas ferramentas do Sistema X Window que um administrador precisa conhecer para configurar o X.

O X Window System é multiplataforma e roda em diversos sistemas operacionais, como Linux, BSDs, Solaris e outros sistemas semelhantes ao Unix. Existem também implementações disponíveis para o macOS da Apple e o Microsoft Windows..

A versão principal do protocolo X usado nas distribuições Linux modernas é X.org versão 11, comumente escrita como *X11*. O protocolo X é o mecanismo de comunicação entre o cliente X e o servidor X. As diferenças entre o cliente X e o servidor X serão discutidas abaixo.

NOTE

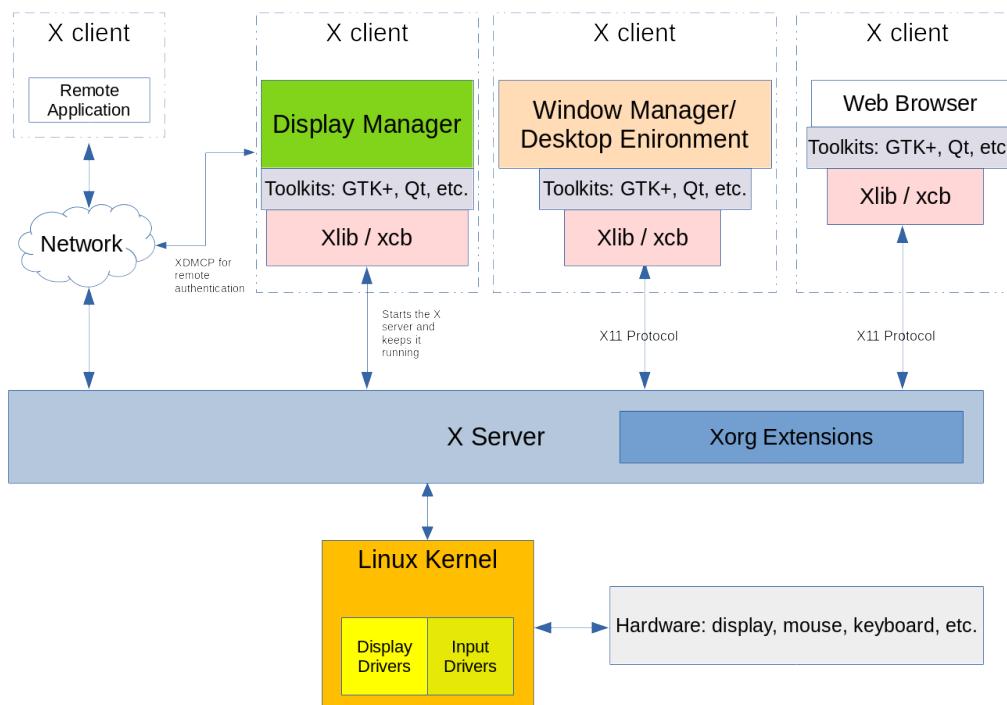
O predecessor do X Window System foi um sistema de janelas chamado *W*, um

esforço de desenvolvimento conjunto entre a IBM, a DEC e o MIT. Esse software foi derivado do *Project Athena* em 1984. Quando os desenvolvedores começaram a trabalhar em um novo servidor de exibição, eles escolheram a letra seguinte do alfabeto inglês: “X”. A evolução do X Window System é atualmente controlada pelo *MIT X Consortium*.

A arquitetura do X Window System

O X Window System fornece os mecanismos para desenhar formas bidimensionais básicas (e formas tridimensionais por meio de extensões) em uma tela. Ele se divide em um cliente e um servidor e, na maioria das instalações em que uma área de trabalho gráfica é necessária, ambos os componentes estão no mesmo computador. O componente cliente assume a forma de um aplicativo, como um emulador de terminal, um game ou um navegador web. Cada aplicativo cliente informa ao servidor X sobre a localização e o tamanho de sua janela na tela do computador. O cliente também decide o que vai para aquela janela, e o servidor X exibe o desenho solicitado na tela. O X Window System também lida com os dados enviados por dispositivos como mouses, teclados, trackpads e muito mais.

Estrutura básica de um X Window System



O X Window System é compatível com rede. Múltiplos clientes X de diferentes computadores de

uma rede podem fazer solicitações de desenho a um único servidor X remoto. A ideia por trás disso é que um administrador ou usuário pode ter acesso a um aplicativo gráfico em um sistema remoto que talvez não esteja disponível em seu sistema local.

Um recurso importante do X Window System é que ele é modular. Ao longo da existência do X Window System, recursos mais novos foram sendo desenvolvidos e adicionados à sua estrutura. Esses novos componentes foram adicionados apenas como extensões ao servidor X, deixando o protocolo central do X11 intacto. Essas extensões estão contidas nos arquivos de biblioteca *Xorg*. Eis alguns exemplos de bibliotecas *Xorg*: *libXrandr*, *libXcursor*, *libX11*, *libxkbfile* e muitas outras que fornecem funcionalidades extras ao servidor X.

O *gerenciador de exibição* fornece um login gráfico para um sistema. Esse sistema pode ser um computador local ou uma máquina em uma rede. O gerenciador de exibição é iniciado após a inicialização do computador e inicia uma sessão do servidor X para o usuário autenticado. O gerenciador de exibição também é responsável por manter o servidor X em funcionamento. Como exemplo de gerenciadores de exibição, podemos citar o GDM, o SDDM e o LightDM.

Cada instância de um servidor X em execução possui um *nome de exibição* que a identifica. O nome de exibição contém o seguinte:

```
hostname:displaynumber.screennumber
```

O nome de exibição também informa a um aplicativo gráfico onde ele deve ser renderizado e em qual hospedeiro (no caso de uma conexão X remota).

O *hostname* refere-se ao nome do sistema que exibirá o aplicativo. Se o nome de exibição não contiver o nome do hospedeiro, o host local será pressuposto.

O *displaynumber* faz referência à coleção de “telas” que estão em uso, seja uma única tela de laptop ou diversas telas em uma estação de trabalho. Cada sessão do servidor X em execução recebe um número de exibição começando em 0.

O *screennumber* padrão é 0. Esse pode ser o caso se apenas uma tela física ou diversas telas físicas estiverem configuradas para funcionar como uma só tela. Quando todas as telas de uma configuração de múltiplos monitores são combinadas em uma única tela lógica, as janelas do aplicativo podem ser movidas livremente entre as telas. Em situações em que cada tela é configurada para funcionar independentemente uma da outra, cada tela abrigará as janelas dos aplicativos que forem abertos dentro delas e as janelas não podem ser movidas de uma tela para outra. A cada tela independente será atribuído seu próprio número. Se houver apenas uma tela lógica em uso, o ponto e o número da tela serão omitidos.

O nome de exibição de uma sessão X em execução é armazenado na variável de ambiente `DISPLAY`:

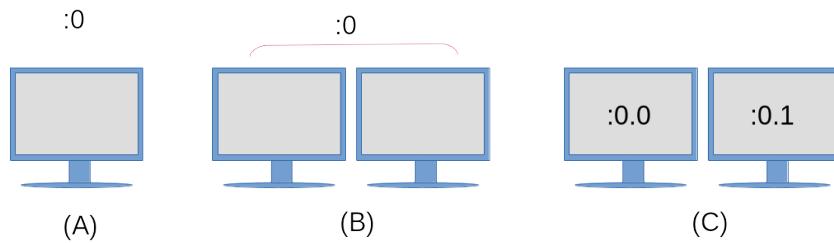
```
$ echo $DISPLAY
:0
```

A saída detalha o seguinte:

1. O servidor X em uso está no sistema local, portanto não há nada impresso à esquerda dos dois pontos.
2. A sessão atual do servidor X é a primeira indicada por `:0` imediatamente após os dois pontos.
3. Há apenas uma tela lógica em uso, portanto um número de tela não é visível.

Para ilustrar melhor esse conceito, consulte o seguinte diagrama:

Exemplos de configuração de exibição



(A)

Um único monitor, com uma única configuração de monitor e apenas uma tela.

(B)

Configurado como um único monitor, com dois monitores físicos configurados como uma tela. As janelas do aplicativo podem ser movidas livremente entre os dois monitores.

(C)

Uma configuração de tela única (conforme indicado por `:0`), porém cada monitor é uma tela independente. Ambas as telas ainda compartilharão os mesmos dispositivos de entrada, como teclado e mouse; porém, um aplicativo aberto na tela `:0.0` não pode ser movido para a tela `:0.1` e vice-versa.

Para iniciar um aplicativo em uma tela específica, atribua o número da tela à variável de ambiente `DISPLAY` antes de iniciar o aplicativo: \$

```
$ DISPLAY=:0.1 firefox &
```

Esse comando iniciaria o navegador Firefox na tela à direita do diagrama acima. Alguns kits de ferramentas também oferecem opções de linha de comando para instruir um aplicativo a ser executado em uma tela especificada. Procure por `--screen` end `--display` na página do manual de `gtk-options(7)` para ver um exemplo..

Configuração do servidor X

Tradicionalmente, o principal arquivo de configuração usado para configurar um servidor X é o arquivo `/etc/X11/xorg.conf`. Nas distribuições Linux modernas, o servidor X configura a si mesmo em tempo de execução quando é iniciado e, portanto, nenhum arquivo `xorg.conf` pode existir.

O arquivo `xorg.conf` é dividido em estrofes separadas chamadas *seções*. Cada seção começa com o termo `Section` e, após este termo está o *nome da seção*, que se refere à configuração de um componente. Cada `Section` é encerrada por uma `EndSection` correspondente. Um arquivo `xorg.conf` típico contém as seguintes seções:

InputDevice

Usada para configurar um modelo específico de teclado ou mouse.

InputClass

`InputClass` Nas distribuições Linux modernas, esta seção é tipicamente encontrada em um arquivo de configuração à parte, localizado em `/etc/X11/xorg.conf.d/`. `InputClass` é usada para configurar uma classe de dispositivos de hardware como teclados e mouses, e não um componente específico de hardware. Veja abaixo um exemplo de arquivo `/etc/X11/xorg.conf.d/00-keyboard.conf`:

```
Section "InputClass"
    Identifier "system-keyboard"
    MatchIsKeyboard "on"
    Option "XkbLayout" "us"
    Option "XkbModel" "pc105"
EndSection
```

A opção de `XkbLayout` determina a disposição das teclas de um teclado, como Dvorak, canhoto ou destro, QWERTY e idioma. A opção de `XkbModel` é usada para definir o tipo de teclado utilizado. Há uma tabela de modelos, layouts e suas descrições em `xkeyboard-config(7)`. Os

arquivos associados aos layouts de teclado podem ser encontrados em `/usr/share/X11/xkb`. Um layout de teclado grego politônico em um computador Chromebook apareceria desta maneira:

```
Section "InputClass"
    Identifier "system-keyboard"
    MatchIsKeyboard "on"
    Option "XkbLayout" "gr(polytonic)"
    Option "XkbModel" "chromebook"
EndSection
```

Alternativamente, o layout de um teclado pode ser modificado durante uma sessão X em execução com o comando `setxkbmap`. Eis um exemplo desse comando para configurar o layout grego politônico em um computador Chromebook:

```
$ setxkbmap -model chromebook -layout "gr(polytonic)"
```

Essa configuração só permanecerá ativa enquanto a sessão X estiver em uso. Para que essas alterações se tornem permanentes, modifique o arquivo `/etc/X11/xorg.conf.d/00-keyboard.conf` de forma a incluir as configurações necessárias.

NOTE O comando `setxkbmap` utiliza a X Keyboard Extension (XKB). Este é um exemplo da funcionalidade aditiva do X Window System por meio do uso de extensões.

As distribuições Linux modernas fornecem o comando `localectl` através do `systemd`, que também pode ser usado para modificar um layout de teclado e cria automaticamente o arquivo de configuração `/etc/X11/xorg.conf.d/00-keyboard.conf`. Novamente, eis um exemplo de configuração de um teclado grego politônico em um Chromebook, desta vez com o comando `localectl`:

```
$ localectl --no-convert set-x11-keymap "gr(polytonic)" chromebook
```

A opção `--no-convert` é usada aqui para impedir que o `localectl` modifique o mapa do teclado no console do hospedeiro.

Monitor

A seção `Monitor` descreve o monitor físico utilizado e onde está conectado. Eis um exemplo de configuração que mostra um monitor de hardware conectado à segunda porta de vídeo e usado como monitor principal.

```
Section "Monitor"
    Identifier  "DP2"
    Option      "Primary" "true"
EndSection
```

Device

A seção `Device` descreve a placa de vídeo física utilizada. A seção também contém o módulo do kernel usado como driver para a placa de vídeo, junto com sua localização física na placa-mãe.

```
Section "Device"
    Identifier  "Device0"
    Driver      "i915"
    BusID       "PCI:0:2:0"
EndSection
```

Screen

A seção `Screen` reúne as seções `Monitor` e `Device`. Um exemplo de seção `Screen` seria semelhante ao seguinte:

```
Section "Screen"
    Identifier "Screen0"
    Device     "Device0"
    Monitor   "DP2"
EndSection
```

ServerLayout

A seção `ServerLayout` agrupa todas as seções como mouse, teclado e telas em uma única interface do X Window System.

```
Section "ServerLayout"
    Identifier "Layout-1"
    Screen     "Screen0" 0 0
    InputDevice "mouse1"  "CorePointer"
    InputDevice "system-keyboard" "CoreKeyboard"
EndSection
```

NOTE Nem todas as seções estão presentes em um arquivo de configuração. Nos casos em que uma seção está ausente, os valores padrão são fornecidos pela instância do servidor X em execução.

Os arquivos de configuração específicos ao usuário também residem em `/etc/X11/xorg.conf.d/`. Os arquivos de configuração fornecidos pela distribuição localizam-se em `/usr/share/X11/xorg.conf.d/`. Os arquivos de configuração localizados em `/etc/X11/xorg.conf.d/` são analisados antes do arquivo `/etc/X11/xorg.conf` se ele existir no sistema.

O comando `xdisplayinfo` é usado em um computador para exibir informações sobre uma instância do servidor X em execução. Veja abaixo um exemplo de saída do comando:

```
$ xdisplayinfo
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
  ...

```

As partes mais relevantes da saída estão em negrito, como o nome da tela (que é idêntico ao conteúdo da variável de ambiente DISPLAY), as informações de versão do servidor X em uso, o número e a listagem das extensões do Xorg em uso e mais informações sobre a tela em si.

Criando um arquivo de configuração básico do Xorg

Mesmo que nas instalações modernas do Linux o X crie sua configuração após a inicialização do sistema, um arquivo `xorg.conf` ainda pode ser usado. Para gerar um arquivo

/etc/X11/xorg.conf permanente, execute o seguinte comando:

```
$ sudo Xorg -configure
```

Se já houver uma sessão X em execução, será necessário especificar um DISPLAY diferente em seu comando, por exemplo:

NOTE

```
$ sudo Xorg :1 -configure
```

Em algumas distribuições Linux, o comando X pode ser usado no lugar de Xorg, pois X é um link simbólico para Xorg.

Um arquivo xorg.conf.new será criado em seu diretório de trabalho atual. O conteúdo desse arquivo deriva-se do que o servidor X descobriu estar disponível em hardware e drivers no sistema local. Para usar esse arquivo, ele deve ser movido para o diretório /etc/X11/ e renomeado como `xorg.conf`:

```
$ sudo mv xorg.conf.new /etc/X11/xorg.conf
```

NOTE

As seguintes páginas de manual fornecem mais informações sobre os componentes do X Window System: xorg.conf(5), Xserver(1), X(1) and Xorg(1).

Wayland

Wayland é o protocolo de exibição mais recente, criado para substituir o X Window System. Ele é o servidor de exibição padrão de muitas distribuições Linux modernas. Foi projetado para usar menos recursos do sistema e ocupar um espaço de instalação menor do que o X. O projeto foi iniciado em 2010 e ainda está em desenvolvimento ativo, com a colaboração de desenvolvedores e ex-desenvolvedores do X.org.

Ao contrário do X Window System, não existe uma instância de servidor executada entre o cliente e o kernel. Em vez disso, uma janela do cliente trabalha com seu próprio código ou o código de um kit de ferramentas (como Gtk+ ou Qt) para realizar a renderização. Para que a renderização seja feita, uma solicitação é enviada ao kernel do Linux por meio do protocolo Wayland. O kernel encaminha a solicitação através do protocolo Wayland para o *compositor* Wayland, que gerencia os dados inseridos pelo dispositivo, o gerenciamento de janelas e a composição. O compositor é a parte do sistema que combina os elementos renderizados em uma saída visual na tela.

A maioria dos kits de ferramentas modernos, como o Gtk+ 3 e o Qt 5, foram atualizados para

permitir a renderização para um sistema X Window ou um computador executando o Wayland. Por enquanto, nem todos os aplicativos autônomos foram escritos para suportar a renderização no Wayland. Para aplicativos e estruturas que ainda precisam do X Window System para rodar, o aplicativo pode ser executado dentro do *XWayland*. O sistema XWayland é um servidor X separado que roda em um cliente Wayland e, portanto, renderiza o conteúdo da janela do cliente em uma instância independente do servidor X.

Assim como o X Window System usa uma variável de ambiente `DISPLAY` para controlar as telas em uso, o protocolo Wayland usa uma variável de ambiente `WAYLAND_DISPLAY`. Veja abaixo um exemplo de saída de um sistema executando um monitor Wayland:

```
$ echo $WAYLAND_DISPLAY  
wayland-0
```

Esta variável de ambiente não está disponível em sistemas que utilizam X.

Exercícios Guiados

1. Qual comando você usaria para determinar quais extensões Xorg estão disponíveis em um sistema?

2. Você acaba de receber um mouse de 10 botões novinho em folha para o seu computador, porém ele necessita de uma configuração extra para que todos os botões funcionem corretamente. Sem modificar o resto da configuração do servidor X, que diretório você usaria para criar um novo arquivo de configuração para este mouse e qual seção específica da configuração seria usada neste arquivo?

3. Qual componente de uma instalação do Linux é responsável por manter um servidor X funcionando?

4. Qual opção de linha de comando é usada com o comando X para criar um novo arquivo de configuração `xorg.conf`?

Exercícios Exploratórios

1. Qual seria o conteúdo da variável de ambiente DISPLAY em um sistema chamado lab01 usando uma configuração de tela única? Suponha que a variável de ambiente DISPLAY esteja sendo visualizada em um emulador de terminal na terceira tela independente.

2. Que comando pode ser usado para criar um arquivo de configuração de teclado para uso pelo X Window System?

3. Em uma instalação típica do Linux, um usuário pode alternar para um terminal virtual pressionando as teclas `Ctrl + Alt + F1-F6` em um teclado. Você recebeu a missão de configurar um sistema de quiosque com uma interface gráfica e precisa desabilitar esse recurso para evitar uma adulteração não autorizada do sistema. Você decide criar um arquivo de configuração `/etc/X11/xorg.conf.d/10-kiosk.conf`. Usando uma seção `ServerFlags` (usada para definir as opções globais do Xorg no servidor), qual opção precisaria ser especificada? Consulte a página do manual do `xorg(1)` para localizar a opção.

Resumo

Esta lição tratou do X Window System e seu uso no Linux. O X Window System é usado para desenhar imagens e texto nas telas, conforme definido em diversos arquivos de configuração. O X Window System é freqüentemente usado para configurar dispositivos de entrada como mouses e teclados. Esta lição discutiu os seguintes pontos:

- A arquitetura do X Window System em alto nível.
- Quais arquivos de configuração são usados para configurar um sistema X Window e sua localização no sistema de arquivos.
- Como usar a variável de ambiente DISPLAY em um sistema rodando X.
- Uma breve introdução ao protocolo de exibição Wayland.

Os comandos e arquivos de configuração abordados foram:

- Modificação do layout do teclado dentro de uma instalação do Xorg com `setxkbmap` e `localectl`.
- O comando `Xorg` para criar um novo arquivo de configuração `/etc/X11/xorg.conf`.
- O conteúdo dos arquivos de configuração do Xorg localizados em: `/etc/X11/xorg.conf`, `/etc/X11/xorg.conf.d/` and `/usr/share/X11/xorg.conf.d/`.
- O comando `xpyinfo` para exibir informações gerais sobre uma sessão do servidor X em execução.

Respostas aos Exercícios Guiados

1. Qual comando você usaria para determinar quais extensões Xorg estão disponíveis em um sistema?

```
$ xdpinfo
```

2. Você acaba de receber um mouse de 10 botões novinho em folha para o seu computador, porém ele necessita de uma configuração extra para que todos os botões funcionem corretamente. Sem modificar o resto da configuração do servidor X, que diretório você usaria para criar um novo arquivo de configuração para este mouse e qual seção específica da configuração seria usada neste arquivo?

As configurações definidas pelo usuário devem estar localizadas em `/etc/X11/xorg.conf.d/` e a seção específica necessária para esta configuração do mouse seria `InputDevice`.

3. Qual componente de uma instalação do Linux é responsável por manter um servidor X funcionando?

O gerenciador de exibição.

4. Qual opção de linha de comando é usada com o comando X para criar um novo arquivo de configuração `xorg.conf`?

```
-configure
```

Lembre-se de que o comando `X` é um link simbólico para o comando `Xorg`.

Respostas aos Exercícios Exploratórios

1. Qual seria o conteúdo da variável de ambiente `DISPLAY` em um sistema chamado `lab01` usando uma configuração de tela única? Suponha que a variável de ambiente `DISPLAY` esteja sendo visualizada em um emulador de terminal na terceira tela independente.

```
$ echo $DISPLAY
lab01:0.2
```

2. Que comando pode ser usado para criar um arquivo de configuração de teclado para uso pelo X Window System?

```
$ localectl
```

3. Em uma instalação típica do Linux, um usuário pode alternar para um terminal virtual pressionando as teclas `Ctrl` + `Alt` + `F1`-`F6` em um teclado. Você recebeu a missão de configurar um sistema de quiosque com uma interface gráfica e precisa desabilitar esse recurso para evitar uma adulteração não autorizada do sistema. Você decide criar um arquivo de configuração `/etc/X11/xorg.conf.d/10-kiosk.conf`. Usando uma seção `ServerFlags` (usada para definir as opções globais do Xorg no servidor), qual opção precisaria ser especificada? Consulte a página do manual do `xorg` (1) para localizar a opção.

```
Section "ServerFlags"
    Option "DontVTSwitch" "True"
EndSection
```



106.2 Desktops gráficos

Referência ao LPI objectivo

[LPIC-1 version 5.0, Exam 102, Objective 106.2](#)

Peso

1

Áreas chave de conhecimento

- Noções dos principais ambientes de desktop
- Noções dos protocolos utilizados para acessar sessões de desktop remoto.

Segue abaixo uma lista parcial dos arquivos, termos e utilitários usados

- KDE
- Gnome
- Xfce
- X11
- XDMCP
- VNC
- Spice
- RDP



**Linux
Professional
Institute**

106.2 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	106 Interfaces de usuário e áreas de trabalho
Objetivo:	106.2 Áreas de trabalho gráficas
Lição:	1 de 1

Introdução

Os sistemas operacionais baseados em Linux são conhecidos por sua avançada interface de linha de comando, mas ela pode assustar um pouco os usuários sem uma formação mais técnica. Para tornar o uso do computador mais intuitivo, a combinação de telas de alta resolução com dispositivos apontadores deu origem a interfaces de usuário baseadas em imagens. Enquanto a interface de linha de comando exige um conhecimento prévio dos nomes dos programas e suas opções de configuração, a funcionalidade dos programas com *interface gráfica do usuário* (GUI) pode ser acionada simplesmente clicando-se em elementos visuais familiares, o que torna a curva de aprendizado bem mais suave. Além disso, a interface gráfica é mais adequada para tarefas de multimídia e outras atividades visualmente orientadas.

De fato, a interface gráfica de usuário tornou-se quase um sinônimo de interface de computação. A maioria das distribuições Linux vem com uma interface gráfica instalada por padrão. Não existe, entretanto, um único programa monolítico responsável pelos desktops gráficos completos nos sistemas Linux. Em vez disso, cada área de trabalho gráfica compõe-se, na verdade, de uma grande coleção de programas e suas dependências, que variam segundo as escolhas da própria distribuição ou o gosto pessoal do usuário.

X Window System

No Linux e em outros sistemas operacionais semelhantes ao Unix, o *X Window System* (também conhecido como *X11* ou apenas *X*) fornece os recursos de baixo nível relacionados à renderização da interface gráfica e à interação do usuário com ela, como:

- O tratamento dos eventos de entrada, como movimentos do mouse ou pressionamentos de teclas.
- A capacidade de cortar, copiar e colar conteúdo de texto entre aplicativos separados.
- A interface de programação usada por outros programas para desenhar os elementos gráficos.

Embora o X Window System seja responsável por controlar a exibição gráfica (o driver de vídeo em si é parte do X), ele não se destina a desenhar elementos visuais complexos por conta própria. Formas, cores, sombras e outros efeitos visuais são gerados pelo aplicativo que roda em cima do X. Esta abordagem dá mais liberdade aos aplicativos para criar interfaces personalizadas, mas também pode levar a sobrecargas de desenvolvimento que vão além do escopo do aplicativo, além de inconsistências na aparência e no comportamento em relação a outras interfaces de programas.

Do ponto de vista do desenvolvedor, a introdução do *ambiente de trabalho* (ou ambiente de desktop) facilita a programação da GUI durante o desenvolvimento do aplicativo, ao passo que, da perspectiva do usuário, ele oferece uma experiência consistente entre aplicativos distintos. Os ambientes de desktop reúnem interfaces de programação, bibliotecas e programas de suporte que cooperam para entregar conceitos de design tradicionais, mas em constante evolução.

Ambiente de trabalho

A GUI tradicional consiste em diversas janelas — o termo *janela* refere-se aqui a qualquer área autônoma da tela — associadas aos processos em execução. Como o X Window System sozinho oferece apenas recursos interativos básicos, a experiência completa do usuário depende dos componentes fornecidos pelo ambiente de área de trabalho.

O componente mais importante de um ambiente de área de trabalho é provavelmente o *gerenciador de janelas*, que controla o posicionamento e as decorações das janelas. É o gerenciador de janelas que adiciona a barra de título à janela, bem como os botões de controle — geralmente associados às ações de minimizar, maximizar e fechar — e gerencia a alternância entre as janelas abertas.

NOTE

Os conceitos básicos empregados na interface gráfica dos computadores pessoais vieram de ideias tiradas de espaços de trabalho de escritório de verdade.

Metaforicamente falando, a tela do computador é a escrivaninha na qual colocamos objetos como documentos e pastas. Uma janela de aplicativo com o conteúdo de um documento simula atos físicos, como preencher um formulário ou pintar uma imagem. Como as áreas de trabalho reais, os desktops de computador também contêm acessórios de software como blocos de notas, relógios, calendários, etc., a maioria deles baseados em seus equivalentes “da vida real”.

Todos os ambientes de desktop incluem um gerenciador de janelas cuja aparência e usabilidade se baseia em um *kit de ferramentas de widget*. Os widgets são elementos visuais informativos ou interativos, como botões ou campos de entrada de texto, distribuídos dentro da janela do aplicativo. Os componentes padrão da área de trabalho — como o inicializador de aplicativos, a barra de tarefas, etc. — e o próprio gerenciador de janelas dependem desses kits de ferramentas de widget para estruturar suas interfaces.

As bibliotecas de software, como o *GTK+* e o *Qt*, fornecem widgets que os programadores podem usar para construir interfaces gráficas elaboradas para seus aplicativos. Historicamente, os aplicativos desenvolvidos com *GTK+* não se assemelhavam aos aplicativos feitos com *Qt* e vice-versa, mas o melhor suporte a temas dos ambientes de desktop de hoje torna a distinção menos óbvia.

No geral, o *GTK+* e o *Qt* oferecem os mesmos recursos em relação aos widgets. Os elementos interativos simples podem ser indistinguíveis, enquanto os widgets compostos — como a janela de diálogo usada por aplicativos para abrir ou salvar arquivos — podem, no entanto, ter uma aparência bem diferenciada. No entanto, aplicativos construídos com kits de ferramentas distintos podem ser executados simultaneamente, independentemente do kit de ferramentas de widget usado pelos outros componentes da área de trabalho.

Além dos componentes básicos da área de trabalho, que podem ser considerados programas individuais por si sós, os ambientes de área de trabalho seguem a metáfora da escrivaninha, fornecendo um conjunto mínimo de programas acessórios desenvolvidos a partir das mesmas diretrizes de design. Todos os principais ambientes de desktop propõem variações dos seguintes aplicativos:

Aplicativos relacionados ao sistema

Emulador de terminal, gerenciador de arquivos, gerenciador de instalação de pacotes, ferramentas de configuração do sistema.

Comunicação e Internet

Gerenciador de contatos, cliente de email, navegador web.

Aplicativos de escritório

Calendário, calculadora, editor de texto.

Os ambientes de desktop podem incluir muitos outros serviços e aplicativos: a tela de login inicial, o gerenciador de sessão, a comunicação entre processos, o chaveiro, etc. Eles também incorporam recursos fornecidos por serviços de terceiros, como o *PulseAudio* para som e o *CUPS* para impressão. Esses recursos não necessitam do ambiente gráfico para funcionar, mas o ambiente de trabalho fornece interfaces gráficas que facilitam sua configuração e operação.

Ambientes de trabalho populares

Muitos sistemas operacionais proprietários suportam oficialmente apenas um único ambiente de desktop, que é vinculado àquela versão específica e não deve ser alterado. Já os sistemas operacionais baseados em Linux suportam diferentes opções de ambiente de trabalho que podem ser usadas em conjunto com o X. Cada ambiente de área de trabalho tem seus próprios recursos, mas eles geralmente compartilham alguns conceitos de design:

- Um inicializador de aplicativos que lista os aplicativos internos e de terceiros disponíveis no sistema.
- Regras que definem os aplicativos padrão associados a tipos de arquivos e protocolos.
- Ferramentas de configuração para personalizar a aparência e o comportamento do ambiente de trabalho.

O *Gnome* é um dos ambientes de desktop mais populares, sendo a primeira escolha em distribuições como Fedora, Debian, Ubuntu, SUSE Linux Enterprise, Red Hat Enterprise Linux, CentOS, etc. Em sua versão 3, o Gnome recebeu uma grande repaginada em sua aparência e estrutura, afastando-se da metáfora da escrivaninha e apresentando o *Gnome Shell* como sua nova interface.

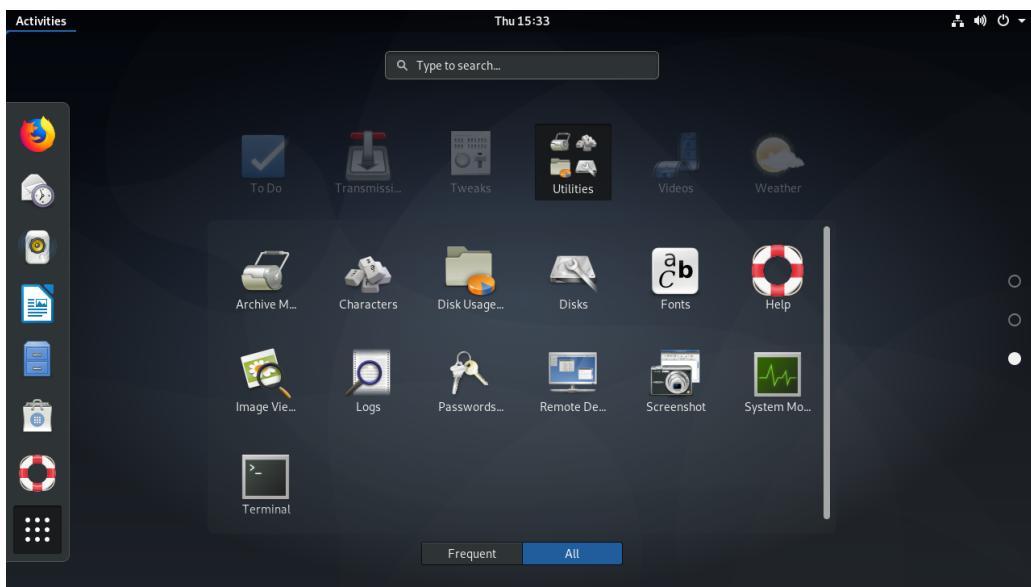


Figure 1. Gnome Shell Activities

O iniciador de tela cheia de uso geral *Gnome Shell Activities* substituiu o iniciador de aplicativos e a barra de tarefas tradicionais. No entanto, ainda é possível usar o Gnome 3 com a aparência antiga escolhendo a opção *Gnome Classic* na tela de login.

O KDE é um grande ecossistema de aplicativos e plataforma de desenvolvimento. Sua última versão para o ambiente de desktop, *KDE Plasma*, é usada por padrão no openSUSE, Mageia, Kubuntu, etc. O emprego da biblioteca Qt é um recurso marcante do KDE, emprestando-lhe sua aparência inconfundível e uma infinidade de aplicativos originais. O KDE fornece, ainda, uma ferramenta de configuração para garantir a coesão visual com os aplicativos GTK+.

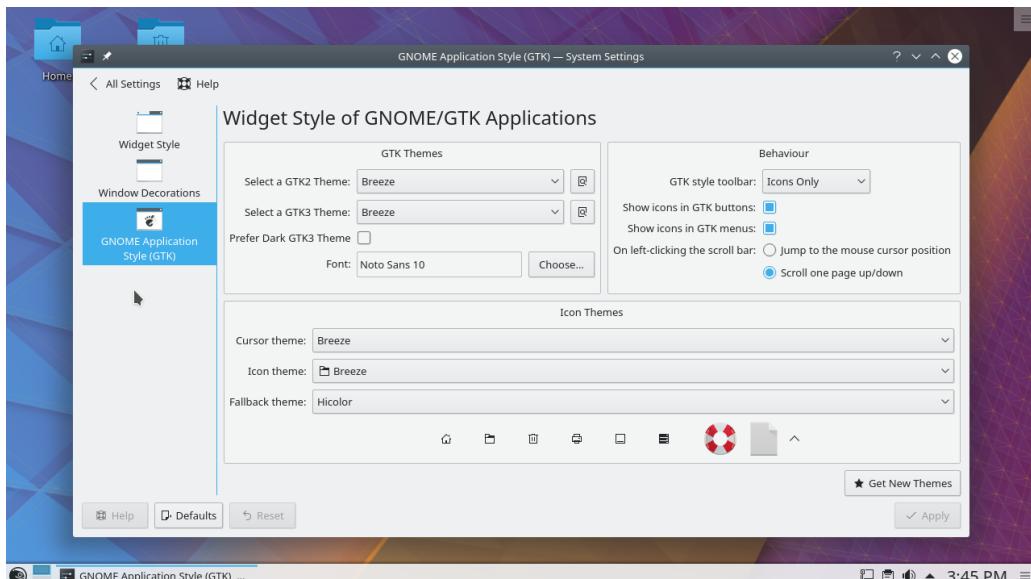


Figure 2. Configurações GTK do KDE

O *Xfce* é um ambiente de desktop que visa ser esteticamente agradável sem consumir muitos recursos da máquina. Sua estrutura é altamente modularizada, permitindo ao usuário ativar e desativar componentes de acordo com suas necessidades e preferências.

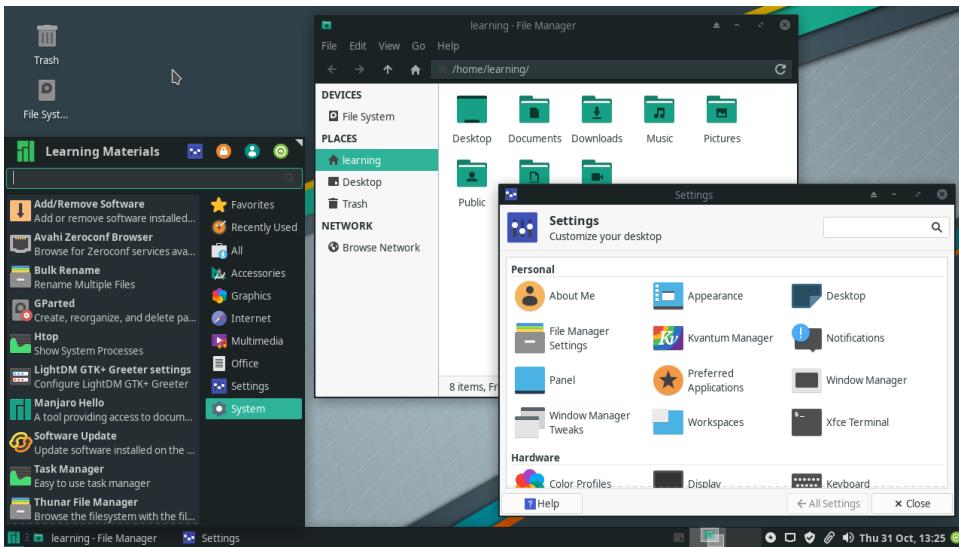


Figure 3. A área de trabalho do *Xfce*

Existem muitos outros ambientes de desktop para Linux, geralmente fornecidos por versões alternativas das distribuições. A distribuição Linux Mint, por exemplo, propõe dois ambientes de trabalho originais: o *Cinnamon* (um fork do Gnome 3) e o *MATE* (um fork do Gnome 2). O *LXDE* é um ambiente de desktop adaptado para baixo consumo de recursos, sendo assim uma boa escolha para instalação em equipamentos mais antigos ou computadores de placa única. Embora não ofereça todas as capacidades dos ambientes de trabalho mais pesados, o *LXDE* inclui todos os recursos básicos esperados de uma interface gráfica de usuário moderna.

TIP

Os atalhos de teclado desempenham um papel importante na interação com o ambiente de trabalho. Alguns atalhos de teclado — como `Alt + Tab` para alternar entre janelas ou `Ctrl + C` para copiar texto — são universais nos ambientes de desktop, mas cada ambiente também possui seus próprios atalhos de teclado. Os atalhos de teclado são encontrados na ferramenta de configuração do teclado fornecida pelo ambiente de desktop, onde os atalhos podem ser adicionados ou modificados.

Interoperabilidade da área de trabalho

A diversidade dos ambientes de desktop nos sistemas operacionais baseados em Linux impõe um desafio: como fazê-los funcionar corretamente com aplicativos gráficos de terceiros ou serviços de sistema sem ser preciso implementar um suporte específico a cada um deles? O compartilhamento de métodos e especificações entre ambientes de desktop melhoraram muito a experiência do usuário e resolve muitos problemas de desenvolvimento, já que os aplicativos

gráficos precisam interagir com o ambiente de desktop atual, independentemente daquele para o qual foram originalmente projetados. Além disso, é importante manter as configurações gerais da área de trabalho caso o usuário decida alterar sua escolha de ambiente.

Um grande conjunto de especificações para a interoperabilidade de áreas de trabalho é mantido pela organização *freedesktop.org*. A adoção da especificação completa não é obrigatória, mas muitas regras são amplamente utilizadas:

Locais dos diretórios

Onde as configurações pessoais e outros arquivos específicos do usuário estão localizados.

Itens da área de trabalho

Os aplicativos de linha de comando podem ser executados no ambiente da área de trabalho por meio de qualquer emulador de terminal, mas seria muito confuso disponibilizar todos eles no inicializador de aplicativos. Os itens da área de trabalho são arquivos de texto que terminam com `.desktop`, usados pelo ambiente de desktop para reunir informações sobre os aplicativos de área de trabalho disponíveis e como usá-los.

Início automático de aplicativos

Itens da área de trabalho que indicam quais aplicativos devem ser iniciados automaticamente após o usuário efetuar login.

Arrastar e soltar

Como os aplicativos devem lidar com eventos de arrastar e soltar.

Lixeira

A localização comum de arquivos excluídos pelo gerenciador de arquivos, bem como os métodos para armazenar e remover arquivos de lá.

Temas de ícones

O formato comum para bibliotecas de ícones intercambiáveis.

A facilidade de uso oferecida pelos ambientes de desktop tem uma desvantagem em comparação com as interfaces de texto como o shell: a capacidade de fornecer acesso remoto. Embora o ambiente shell de linha de comando de uma máquina remota possa ser facilmente acessado com ferramentas como `ssh`, o acesso remoto a ambientes gráficos requer métodos diferentes e o desempenho nem sempre é satisfatório em conexões mais lentas.

Acesso não-local

O X Window System adota um design baseado em *displays* (telas) autônomos, nos quais o mesmo

X display manager (gerenciador de exibição X) pode controlar mais de uma sessão de desktop gráfico ao mesmo tempo. Em essência, um display é análogo a um terminal de texto: ambos se referem a uma máquina ou aplicativo de software usado como um ponto de entrada para estabelecer uma sessão independente do sistema operacional. Embora a configuração mais comum envolva uma única sessão gráfica em execução na máquina local, outras configurações menos convencionais também são possíveis:

- Alternar entre sessões de desktop gráfico ativas na mesma máquina.
- Mais de um conjunto de dispositivos de exibição (por exemplo, tela, teclado, mouse) conectados à mesma máquina, cada um controlando sua própria sessão de desktop gráfico.
- Sessões remotas de desktop gráfico, para as quais a interface gráfica é enviada através da rede para um monitor remoto.

As sessões de desktop remoto são suportadas nativamente pelo X, que emprega o *X Display Manager Control Protocol* (XDMCP) para se comunicar com monitores remotos. Devido ao alto uso de largura de banda, o XDMCP raramente é usado através da Internet ou em LANs de baixa velocidade. Também há questões de segurança com o XDMCP: o display local se comunica com um gerenciador de exibição X remoto privilegiado para executar procedimentos remotos; portanto, uma eventual vulnerabilidade possibilitaria a execução de comandos privilegiados arbitrários na máquina remota.

Além disso, o XDMCP requer instâncias X em execução em ambas as extremidades da conexão, o que pode inviabilizar seu uso se o X Window System não estiver disponível em todas as máquinas envolvidas. Na prática, outros métodos mais eficientes e menos invasivos são usados para estabelecer sessões remotas de desktop gráfico.

O *Virtual Network Computing* (VNC) é uma ferramenta não vinculada a uma plataforma que permite visualizar e controlar ambientes de desktop remotos usando o protocolo *Remote Frame Buffer* (RFB). Por meio dele, os eventos produzidos pelo teclado e mouse locais são transmitidos para a área de trabalho remota, que por sua vez envia de volta quaisquer atualizações de tela para serem exibidas localmente. É possível executar muitos servidores VNC na mesma máquina, mas cada servidor VNC precisa de uma porta TCP exclusiva na interface de rede que aceite solicitações de sessão de entrada. Por convenção, o primeiro servidor VNC deve usar a porta TCP 5900, o segundo deve usar 5901 e assim por diante.

O servidor VNC não precisa de privilégios especiais para ser executado. Um usuário comum pode, por exemplo, fazer login em sua conta remota e iniciar seu próprio servidor VNC a partir de lá. Assim, na máquina local, qualquer aplicativo cliente VNC pode ser usado para acessar a área de trabalho remota (supondo que as portas de rede correspondentes sejam alcançáveis). O arquivo `~/.vnc/xstartup` é um script de shell executado pelo servidor VNC quando ele inicia e pode ser

usado para definir qual ambiente de desktop o servidor VNC disponibilizará para o cliente VNC. É importante observar que nativamente o VNC não propõe métodos nativos de criptografia e autenticação modernos, e por isso deve ser usado em conjunto com um aplicativo de terceiros que forneça esses recursos. Com frequência, usam-se métodos que envolvem túneis VPN e SSH para proteger as conexões VNC.

O *Remote Desktop Protocol* (RDP) é usado sobretudo para acessar remotamente a área de trabalho de um sistema operacional *Microsoft Windows* por meio da porta de rede TCP 3389. Embora ele empregue o protocolo RDP proprietário da Microsoft, a implementação do cliente usada nos sistemas Linux consiste em programas de código aberto licenciados sob a *GNU General Public License* (GPL) e não tem restrições legais de uso.

O *Simple Protocol for Independent Computing Environments* (Spice) comprehende um conjunto de ferramentas destinadas a acessar o ambiente de trabalho de sistemas *virtualizados*, seja na máquina local ou em um local remoto. Além disso, o protocolo Spice oferece recursos nativos para integrar os sistemas locais e remotos, como a capacidade de acessar dispositivos locais (por exemplo, os alto-falantes e os dispositivos USB conectados) da máquina remota e compartilhamento de arquivos entre os dois sistemas.

Existem comandos de cliente específicos para se conectar a cada um desses protocolos de desktop remoto, mas o cliente de desktop remoto *Remmina* fornece uma interface gráfica integrada que facilita o processo de conexão, com a opção de armazenar as configurações de conexão para uso posterior. O Remmina tem plugins para cada protocolo individual e existem plugins para XDMCP, VNC, RDP e Spice. A escolha da ferramenta certa depende dos sistemas operacionais envolvidos, da qualidade da conexão de rede e dos recursos do ambiente de área de trabalho remoto que devem estar disponíveis.

Exercícios Guiados

1. Que tipo de aplicativo propõe sessões de shell em janelas no ambiente de desktop?

2. Devido à variedade de ambientes de desktop Linux, o mesmo aplicativo pode ter mais de uma versão, sendo cada uma delas adaptada a um determinado kit de ferramentas de widget. Por exemplo, o cliente bittorrent *Transmission* tem duas versões: *transmission-gtk* e *transmission-qt*. Qual dos dois deve ser instalado para garantir a integração máxima com o KDE?

3. Quais ambientes de desktop Linux são recomendados para computadores de baixo custo, de placa única e com pouco poder de processamento?

Exercícios Exploratórios

1. Existem duas maneiras de copiar e colar texto no Sistema X Window: usando os tradicionais atalhos de teclado `Ctrl + C` e `Ctrl + V` (também disponíveis no menu da janela) ou pressionar o botão do meio do mouse para colar o texto atualmente selecionado. Qual a maneira apropriada de copiar e colar texto de um emulador de terminal?

2. A maioria dos ambientes de desktop atribuem o atalho de teclado `Alt + F2` para a janela *Executar programa*, onde os programas podem ser executados à maneira de uma linha de comando. No KDE, qual comando executaria o emulador de terminal padrão?

3. Qual protocolo é mais adequado para acessar uma área de trabalho remota do Windows a partir de um ambiente de trabalho Linux?

Resumo

Esta lição é uma visão geral dos desktops gráficos disponíveis para os sistemas Linux. Sozinho, o X Window System fornece apenas recursos de interface simples, de modo que os ambientes de desktop enriquecem a experiência do usuário na interface gráfica de janelas. A lição abrange os seguintes tópicos:

- Conceitos de interface gráfica e X Window System.
- Ambientes de desktop disponíveis para Linux.
- Semelhanças e diferenças entre os ambientes de desktop.
- Como acessar um ambiente de desktop remoto.

Os conceitos e programas abordados foram:

- X Window System.
- Ambientes de trabalho populares: KDE, Gnome, Xfce.
- Protocolos de acesso remoto: XDMCP, VNC, RDP, Spice.

Respostas aos Exercícios Guiados

1. Que tipo de aplicativo propõe sessões de shell em janelas no ambiente de desktop?

Qualquer emulador de terminal como Konsole, Gnome terminal, xterm, etc., dá acesso a uma sessão local de shell interativo.

2. Devido à variedade de ambientes de desktop Linux, o mesmo aplicativo pode ter mais de uma versão, sendo cada uma delas adaptada a um determinado kit de ferramentas de widget. Por exemplo, o cliente bittorrent *Transmission* tem duas versões: *transmission-gtk* e *transmission-qt*. Qual dos dois deve ser instalado para garantir a integração máxima com o KDE?

O KDE é construído sobre a biblioteca Qt, por isso a versão Qt—*transmission-qt*—deve ser instalada.

3. Quais ambientes de desktop Linux são recomendados para computadores de baixo custo, de placa única e com pouco poder de processamento?

Ambientes de desktop básicos que não usam muitos efeitos visuais, como Xfce e LXDE.

Respostas aos Exercícios Exploratórios

1. Existem duas maneiras de copiar e colar texto no Sistema X Window: usando os tradicionais atalhos de teclado `Ctrl + C` e `Ctrl + V` (também disponíveis no menu da janela) ou pressionar o botão do meio do mouse para colar o texto atualmente selecionado. Qual a maneira apropriada de copiar e colar texto de um emulador de terminal?

As sessões de shell interativas atribuem o atalho `Ctrl + C` à interrupção de execução dos programas, por isso o método do botão do meio é o mais recomendado.

2. A maioria dos ambientes de desktop atribuem o atalho de teclado `Alt + F2` para a janela *Executar programa*, onde os programas podem ser executados à maneira de uma linha de comando. No KDE, qual comando executaria o emulador de terminal padrão?

O comando `konsole` executa o emulador de terminal do KDE, mas termos genéricos como `terminal` também funcionam.

3. Qual protocolo é mais adequado para acessar uma área de trabalho remota do Windows a partir de um ambiente de área de trabalho Linux?

O Remote Desktop Protocol (RDP), que é suportado nativamente pelo Windows e pelo Linux.



106.3 Acessibilidade

Referência ao LPI objectivo

[LPIC-1 version 5.0, Exam 102, Objective 106.3](#)

Peso

1

Áreas chave de conhecimento

- Conhecimento básico das configurações visuais e temas.
- Conhecimento básico das tecnologias assistivas.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- Temas de Alto Contraste/Texto Grande.
- Leitor de Tela.
- Display Braille.
- Lente de Aumento.
- Teclado Virtual.
- Teclas de aderência e repetição.
- Teclas de alternância.
- Teclas no mouse.
- Gestos.
- Reconhecimento de fala.



106.3 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	106 Interfaces de usuário e áreas de trabalho
Objetivo:	106.3 Acessibilidade
Lição:	1 de 1

Introdução

O ambiente de desktop do Linux inclui muitas configurações e ferramentas para adaptar a interface do usuário a pessoas com deficiência. Os dispositivos de interface humana comuns—tela, teclado e mouse/touchpad—podem ser reconfigurados individualmente para compensar deficiências visuais ou uma mobilidade reduzida.

É possível, por exemplo, adaptar o esquema de cores da área de trabalho para melhor atender pessoas daltônicas. Além disso, quem sofre de lesões por esforço repetitivo pode tirar proveito de métodos alternativos de digitação e clique.

Alguns desses recursos de acessibilidade são fornecidos pelo próprio ambiente de área de trabalho, como o Gnome ou o KDE, e outros são fornecidos por programas adicionais. Neste último caso, é importante escolher a ferramenta que melhor se integra ao ambiente de trabalho, para ter um auxílio de melhor qualidade.

Configurações de acessibilidade

As principais distribuições Linux incluem mais ou menos os mesmos recursos de acessibilidade,

que podem ser personalizados com um módulo especial no gerenciador de configurações que vem com o ambiente de desktop. O módulo de configurações de acessibilidade é chamado de *Acesso universal* na área de trabalho do Gnome, enquanto que no KDE ele está em *Configurações do sistema, Personalização, Acessibilidade*. Outros ambientes de desktop, como o *Xfce*, também o chamam de *Acessibilidade* em seu gerenciador de configurações gráficas. Porém, de maneira geral eles oferecem um conjunto reduzido de funcionalidades em comparação com o Gnome e o KDE.

O Gnome, por exemplo, pode ser configurado para exibir permanentemente o menu de Acesso Universal no canto superior direito da tela, permitindo ativar rapidamente as opções de acessibilidade. Podemos usá-las, por exemplo, para substituir o alerta sonoro por um alerta visual, permitindo que usuários com deficiência auditiva percebam os alertas do sistema com mais facilidade. Embora o KDE não inclua um menu de acesso rápido semelhante, o recurso de alerta visual também está disponível com o nome de *sino visual*.

Assistente de teclado e mouse

O comportamento padrão do teclado e do mouse pode ser modificado para contornar dificuldades específicas de mobilidade. As combinações de teclas, a taxa de repetição automática de teclas e os pressionamentos de tecla não intencionais representam obstáculos significativos para usuários com mobilidade reduzida das mãos. Para compensar esses contratempos de digitação, existem três recursos de acessibilidade relacionados ao teclado: *Teclas de aderência*, *Teclas de repercussão* e *Teclas lentas*.

O recurso *Teclas de aderência* (Sticky keys), encontrado na seção *Digitação* da configuração de Acesso Universal do Gnome, permite ao usuário digitar atalhos de teclado uma tecla por vez. Quando ativado, combinações de teclas como `Ctrl + C` não precisam ser pressionadas ao mesmo tempo. O usuário pode primeiro pressionar a tecla `Ctrl`, soltá-la e em seguida pressionar a tecla `C`. No KDE, esta opção está na guia *Teclas modificadoras* da janela de configurações de Acessibilidade. O KDE também oferece a opção de *Teclas de bloqueio*: se habilitada, as teclas `Alt`, `Ctrl` e Shift permanecerão “apertadas” se o usuário as pressionar duas vezes, semelhante ao comportamento da tecla Caps lock. Como no caso do recurso Caps lock, o usuário precisará pressionar a tecla correspondente novamente para liberá-la.

O recurso *Teclas de repercussão* (Bounce keys) serve para inibir pressionamentos de tecla não intencionais adicionando um tempo de latência entre eles, ou seja, um novo pressionamento de tecla será aceito somente após um período de tempo especificado desde o pressionamento anterior. Com o recurso de teclas de repercussão, os usuários com tremores nas mãos evitam os efeitos de pressionar uma tecla várias vezes quando pretendem pressioná-la apenas uma vez. No Gnome, este recurso diz respeito apenas às repetições da mesma tecla. Já no KDE são levados em conta os pressionamentos de outras teclas. Ele pode ser encontrado na guia *Filtros do teclado*.

O recurso *Teclas lentas* (Slow keys) também ajuda a evitar toques accidentais nas teclas. Quando ativadas, as teclas lentas exigem que o usuário mantenha a tecla pressionada por um período de tempo especificado antes de ela ser aceita. Dependendo das necessidades do usuário, também pode ser útil ajustar a repetição automática enquanto uma tecla é mantida pressionada, disponível nas configurações do teclado.

Os recursos de acessibilidade de teclas de aderência e teclas lentas podem ser ativados e desativados com *Gestos de ativação* realizados no teclado. No KDE, a opção *Usar gestos para ativar as teclas de aderência e as teclas lentas* deve ser marcada para habilitar os Gestos de Ativação, enquanto no Gnome esse recurso é chamado de *Habilitar por Teclado* na janela de configuração do *Assistente de Digitação*. Assim que os Gestos de Ativação forem habilitados, o recurso de teclas de aderência será ativado pressionando-se a tecla Shift cinco vezes consecutivas. Para ativar o recurso de teclas lentas, a tecla Shift deve ser mantida pressionada por oito segundos consecutivos.

Os usuários que acham mais confortável usar o teclado em vez do mouse ou touchpad podem recorrer aos atalhos de teclado para navegar no ambiente de trabalho. Além disso, um recurso chamado *Mouse por teclado* permite que o usuário controle o próprio ponteiro do mouse com o teclado numérico, presente em teclados de desktop de tamanho normal e em laptops maiores.

O teclado numérico é organizado em uma grade quadrada, de modo que cada número corresponde a uma direção: 2 move o cursor para baixo, 4 move o cursor para a esquerda, 7 move o cursor para o noroeste, etc. Por padrão, o número 5 corresponde ao clique esquerdo do mouse.

Ao passo que no Gnome há apenas um interruptor para habilitar a opção Mouse por teclado na janela de configurações de Acesso universal, no KDE as configurações de Mouse por teclado estão localizadas em *Configurações do sistema, Mouse, Navegação por teclado*. Certas opções, como velocidade e aceleração, podem ser personalizadas.

TIP

As teclas lentas, teclas de aderência, teclas de repercussão e o mouse por teclado são recursos de acessibilidade fornecidos pelo AccessX, um recurso existente na extensão X keyboard do X Window System. As configurações do AccessX também podem ser modificadas na linha de comando, com o comando `xkbset`.

O mouse ou o touchpad podem ser usados para gerar entrada de teclado quando o uso do teclado físico for demasiado desconfortável ou impossível. Se a opção *Teclado virtual* nas configurações de acesso universal do Gnome estiver habilitada, um teclado aparecerá na tela sempre que o cursor estiver em um campo de texto; um novo texto é inserido clicando-se nas teclas com o mouse ou os dedos em uma tela de toque, semelhante ao teclado virtual dos smartphones.

O KDE e outros ambientes de área de trabalho nem sempre trazem o teclado virtual por padrão, mas o pacote *onboard*, que pode ser instalado manualmente, oferece um teclado virtual simples

que pode ser usado em qualquer ambiente de área de trabalho. Após a instalação, ele ficará disponível como um aplicativo normal no inicializador de aplicativos.

O comportamento do ponteiro também pode ser modificado, se o gesto de clicar e arrastar o mouse causar dor ou for impraticável por qualquer outro motivo. Se o usuário não conseguir clicar com o botão do mouse rápido o suficiente para acionar um evento de clique duplo, por exemplo, o intervalo de tempo para pressionar o botão do mouse uma segunda vez para um duplo clique pode ser aumentado em *Preferências do mouse* na janela de configuração do sistema.

Se o usuário não conseguir pressionar um ou mais botões do mouse, os cliques podem ser simulados usando diferentes técnicas. Na seção *Assistência de clique* do *Acesso Universal* do Gnome, a opção *Clique secundário simulado* simula um clique com o botão direito se o usuário pressionar e segurar o botão esquerdo do mouse. Com a opção *Clique flutuante* habilitada, um evento de clique será disparado quando o usuário segurar o mouse sem movê-lo. No KDE, o aplicativo *KMouseTool* fornece esses mesmos recursos para facilitar as ações com o mouse.

Deficiências visuais

Alguns usuários com visão reduzida ainda podem usar a tela do monitor para interagir com o computador. Dependendo das necessidades do usuário, muitos ajustes visuais são possíveis para refinar os detalhes de difícil visualização da área de trabalho gráfica padrão.

A seção *Visão* das configurações de *Acesso universal* do Gnome inclui opções que podem ajudar pessoas com visão reduzida:

Alto contraste

torna as janelas e botões mais visíveis exibindo-os em cores mais contrastadas.

Texto grande

amplia o tamanho padrão da fonte na tela.

Tamanho do cursor

permite escolher um cursor maior e mais fácil de localizar na tela.

Alguns desses ajustes não estão estritamente relacionados aos recursos de acessibilidade e, portanto, podem ser encontrados na seção de aparência do utilitário de configuração fornecido por outros ambientes de desktop. Um usuário que tenha dificuldade em discernir entre os elementos visuais pode escolher um tema de alto contraste para facilitar a identificação de botões, janelas sobrepostas, etc.

Se os ajustes de aparência por si só não forem suficientes para aprimorar a legibilidade, um

programa de ampliação pode ser usado para aumentar o zoom em certas partes da tela. Esse recurso é chamado de *Ampliação* nas configurações de *Acesso universal* do Gnome, onde opções como taxa de ampliação, posição da lupa e ajustes de cor podem ser personalizadas.

No KDE, o programa *KMagnifier* oferece os mesmos recursos, mas está disponível como um aplicativo comum através do inicializador de aplicativos. Outros ambientes de desktop podem fornecer seus próprios ampliadores de tela. O Xfce, por exemplo, permite aumentar e diminuir o zoom da tela girando a roda de rolagem do mouse enquanto a tecla `Alt` está pressionada.

Finalmente, os usuários para os quais a interface gráfica não é uma opção podem usar um *leitor de tela* para interagir com o computador. Independentemente do ambiente de trabalho escolhido, o leitor de tela mais popular para os sistemas Linux é o *Orca*, que normalmente vem instalado por padrão na maioria das distribuições. O Orca gera uma voz sintetizada que descreve os eventos na tela e lê o texto sob o cursor do mouse. O Orca também funciona com *visores braille atualizáveis*, dispositivos especiais que exibem caracteres em braille erguendo pequenos pinos que podem ser sentidos com a ponta dos dedos. Nem todos os aplicativos de desktop são totalmente adaptados para leitores de tela e nem todos os usuários acharão fácil usá-los, por isso é importante fornecer o máximo possível de estratégias de leitura de tela para os usuários terem escolha.

Exercícios Guiados

1. Qual recurso de acessibilidade poderia ajudar um usuário a alternar entre as janelas abertas usando o teclado, considerando que o usuário não consegue pressionar as teclas `Alt` e `Tab` ao mesmo tempo?

2. Como o recurso de acessibilidade *Teclas de repercussão* pode ajudar usuários que sofrem de tremores involuntários nas mãos que dificultam a digitação?

3. Quais são os gestos de ativação mais comuns para o recurso de acessibilidade *Teclas de aderência*?

Exercícios Exploratórios

1. Os recursos de acessibilidade nem sempre são fornecidos por um único aplicativo e podem variar de um ambiente de trabalho para outro. No KDE, qual aplicativo ajuda as pessoas com lesões por esforço repetitivo produzindo cliques do mouse sempre que o cursor estiver parado brevemente?

2. Quais aspectos de aparência do ambiente gráfico podem ser modificados para facilitar a leitura do texto na tela?

3. De que forma o aplicativo *Orca* pode ajudar usuários com deficiência visual a interagir com o ambiente de trabalho?

Resumo

Esta lição cobre os recursos gerais de acessibilidade disponíveis nos sistemas Linux. Todos os principais ambientes de desktop, especialmente o Gnome e o KDE, fornecem diversos aplicativos integrados e de terceiros para auxiliar as pessoas com deficiência visual ou mobilidade reduzida. A lição abrange os seguintes tópicos:

- Como alterar as configurações de acessibilidade.
- Maneiras alternativas de usar o teclado e o mouse.
- Adaptações da área de trabalho para os deficientes visuais.

Os comandos e procedimentos abordados foram:

- Configurações de acessibilidade do teclado: Teclas de aderência, teclas lentas, teclas de repercussão.
- Eventos do mouse gerados artificialmente.
- Teclado virtual.
- Ajustes visuais para melhorar a legibilidade.
- Temas da área de trabalho com alto contraste/letras grandes.
- Ampliadores de tela.
- O leitor de tela Orca.

Respostas aos Exercícios Guiados

1. Qual recurso de acessibilidade poderia ajudar um usuário a alternar entre as janelas abertas usando o teclado, considerando que o usuário não consegue pressionar as teclas `Alt` e `Tab` ao mesmo tempo?

O recurso de teclas de aderência, que permite ao usuário digitar atalhos de teclado uma tecla por vez.

2. Como o recurso de acessibilidade *Teclas de repercussão* pode ajudar usuários que sofrem de tremores involuntários nas mãos que dificultam a digitação?

Com as teclas de repercussão ativadas, um novo pressionamento de tecla será aceito somente após um determinado período de tempo desde o pressionamento anterior.

3. Quais são os gestos de ativação mais comuns para o recurso de acessibilidade *Teclas de aderência*?

Se os Gestos de Ativação estiverem habilitados, o recurso teclas de aderência será ativado quando a tecla `Shift` for pressionada cinco vezes consecutivas.

Respostas aos Exercícios Exploratórios

1. Os recursos de acessibilidade nem sempre são fornecidos por um único aplicativo e podem variar de um ambiente de trabalho para outro. No KDE, qual aplicativo ajuda as pessoas com lesões por esforço repetitivo produzindo cliques do mouse sempre que o cursor estiver parado brevemente?

O aplicativo *KMouseTool*.

2. Quais aspectos de aparência do ambiente gráfico podem ser modificados para facilitar a leitura do texto na tela?

Definir um tamanho de fonte maior nas configurações da área de trabalho facilita a leitura de todos os textos na tela.

3. De que forma o aplicativo *Orca* pode ajudar usuários com deficiência visual a interagir com o ambiente de trabalho?

O Orca é um leitor de tela que gera uma voz sintetizada que descreve os eventos na tela e lê o texto sob o cursor do mouse. Ele também funciona com dispositivos chamados de *visores braille atualizáveis*, permitindo ao usuário identificar o texto com padrões tátteis.



Tópico 107: Tarefas administrativas



107.1 Administrar contas de usuário, grupos e arquivos de sistema relacionados

Referência ao LPI objectivo

LPIC-1 version 5.0, Exam 102, Objective 107.1

Peso

5

Áreas chave de conhecimento

- Adicionar, modificar e remover usuários e grupos.
- Gerenciar informações de usuários/grupos em banco de dados senhas/grupos.
- Criar e administrar contas com propósitos especiais e contas limitadas.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- /etc/passwd
- /etc/shadow
- /etc/group
- /etc/skel/
- chage
- getent
- groupadd
- groupdel
- groupmod
- passwd

- `useradd`
- `userdel`
- `usermod`



**Linux
Professional
Institute**

107.1 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	107 Tarefas administrativas
Objetivo:	107.1 Administrar contas de usuários, grupos e arquivos de sistema relacionados
Lição:	1 de 2

Introdução

A administração de usuários e grupos é uma parte fundamental do trabalho de qualquer administrador de sistema. As distribuições Linux modernas incluem interfaces gráficas que permitem gerenciar todas as atividades relacionadas a este aspecto com rapidez e facilidade. Essas interfaces são visualmente diferentes entre si, mas os recursos são os mesmos. Com essas ferramentas, torna-se possível visualizar, editar, adicionar e excluir usuários e grupos locais. No entanto, para tarefas de gerenciamento mais avançadas, é necessário usar a linha de comando.

Adicionando contas de usuário

No Linux, podemos adicionar uma nova conta de usuário com o comando `useradd`. Por exemplo, com privilégios de root, criamos uma nova conta de usuário chamada `michael` com uma configuração padrão, usando o seguinte:

```
# useradd michael
```

Quando você executa o comando `useradd`, as informações do usuário e do grupo armazenadas nos bancos de dados de senha e grupo são atualizadas para a conta de usuário recém-criada e, caso especificado, o diretório inicial do novo usuário também é criado, bem como um grupo com o mesmo nome da nova conta de usuário.

Depois de criar o novo usuário, você pode definir a senha dele usando o comando `passwd`. Para rever o ID de usuário (UID), ID de grupo (GID) e os grupos aos quais esse usuário pertence, usamos os comandos `id` e `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

Lembre-se de que qualquer usuário pode consultar seu UID, GID e os grupos aos quais pertence simplesmente usando os comandos `id` e `groups` sem argumentos, e que qualquer usuário pode alterar sua própria senha com o comando `passwd`. No entanto, apenas os usuários com privilégios de root podem alterar a senha de *qualquer* usuário.

As opções mais importantes que se aplicam ao comando `useradd` são:

-c

Cria uma nova conta de usuário com comentários personalizados (por exemplo, o nome completo do usuário).

-d

Cria uma nova conta de usuário com um diretório pessoal personalizado.

-e

Cria uma nova conta de usuário definindo uma data específica na qual ela será desabilitada.

-f

Cria uma nova conta de usuário definindo o número de dias após a expiração de uma senha durante os quais o usuário deve atualizar a senha (caso contrário, a conta será desabilitada).

-g

Cria uma nova conta de usuário com um GID específico.

-G

Cria uma nova conta de usuário adicionando-a a diversos grupos secundários.

-k

Cria uma nova conta de usuário copiando os arquivos de esqueleto de um diretório personalizado específico (esta opção só é válida se a opção **-m** ou **--create-home** for especificada).

-m

Cria uma nova conta de usuário com seu diretório inicial (se ele não existir).

-M

Cria uma nova conta de usuário sem seu diretório inicial.

-s

Cria uma nova conta de usuário com um shell de login específico.

-u

Cria uma nova conta de usuário com um UID específico.

Veja a lista completa de opções nas páginas de manual do comando `useradd`.

Modificando contas de usuário

Às vezes, precisamos alterar um atributo de uma conta de usuário existente, como o nome de login, o shell de login, a data de expiração da senha e assim por diante. Nesses casos, empregamos o comando `usermod`.

```
# usermod -s /bin/tcsh michael
# usermod -c "Michael User Account" michael
```

Assim como no caso de `useradd`, o comando `usermod` requer privilégios de root.

Nos exemplos acima, o shell de login de `michael` é alterado primeiro e, em seguida, uma breve descrição é adicionada a esta conta de usuário. Lembre-se de que podemos modificar diversos atributos de uma vez, especificando-os em um único comando.

As opções mais importantes que se aplicam ao comando `usermod` são:

-c

Adiciona um breve comentário à conta de usuário especificada.

-d

Altera o diretório inicial da conta de usuário especificada. Quando usado com a opção `-m`, o conteúdo do diretório inicial atual é movido para o novo diretório inicial, que é criado se ainda não existir.

-e

Define a data de expiração da conta de usuário especificada.

-f

Define o número de dias após a expiração da senha durante os quais o usuário deve atualizá-la (caso contrário, a conta é desabilitada).

-g

Muda o grupo primário da conta de usuário especificada (o grupo deve existir).

-G

Adiciona grupos secundários à conta de usuário especificada. Cada grupo deve existir e ser separado do seguinte por uma vírgula, sem espaços em branco intermediários. Quando usada sozinha, esta opção remove todos os grupos existentes aos quais o usuário pertence; quando usada com a opção `-a`, ela simplesmente anexa novos grupos secundários aos existentes.

-l

Altera o nome de login da conta de usuário especificada.

-L

Bloqueia a conta de usuário especificada. Um ponto de exclamação é posto na frente da senha criptografada dentro do arquivo `/etc/shadow`, desabilitando assim o acesso com senha para esse usuário.

-s

Altera o shell de login da conta de usuário especificada.

-u

Altera o UID da conta de usuário especificada.

-U

Desbloqueia a conta de usuário especificada. Remove o ponto de exclamação na frente da senha criptografada no arquivo `/etc/shadow`.

Veja as páginas de manual do comando `usermod` para a lista completa de opções.

TIP Lembre que, ao alterar o nome de login de uma conta de usuário, você provavelmente deve renomear o diretório pessoal desse usuário e outros itens relacionados a ele, como arquivos de spool de email. Lembre também que, ao alterar o UID de uma conta de usuário, provavelmente será preciso corrigir a propriedade dos arquivos e diretórios que estejam fora do diretório inicial do usuário (o ID do usuário é alterado automaticamente na caixa de email do usuário e em todos os arquivos pertencentes ao usuário e localizados no diretório inicial do usuário).

Excluindo contas de usuário

Se quiser excluir uma conta de usuário, você pode usar o comando `userdel`. Em particular, este comando atualiza as informações armazenadas nos bancos de dados de contas, removendo todas as entradas referentes ao usuário especificado. A opção `-r` também remove o diretório pessoal do usuário e todo o seu conteúdo, junto com o spool de email do usuário. Outros arquivos, localizados em outros locais, devem ser buscados e excluídos manualmente.

```
# userdel -r michael
```

Quanto a `useradd` e `usermod`, é necessário ter autoridade de root para excluir contas de usuário.

Adicionando, modificando e excluindo grupos

Como no caso do gerenciamento de usuários, podemos adicionar, modificar e excluir grupos usando os comandos `groupadd`, `groupmod` e `groupdel` com privilégios de root. Se quiser criar um novo grupo chamado `developer`, execute o seguinte comando:

```
# groupadd -g 1090 developer
```

A opção `-g` deste comando cria um grupo com um GID específico.

WARNING

Lembre-se de que quando você adiciona uma nova conta de usuário, o grupo primário e os grupos secundários aos quais ela pertence *precisam* existir antes de iniciar o comando `useradd`.

Mais tarde, se você quiser renomear o grupo `developer` para `web-developer` e alterar seu GID, pode executar o seguinte:

```
# groupmod -n web-developer -g 1050 developer
```

TIP Se você alterar o GID usando a opção `-g`, será necessário alterar o GID de todos os arquivos e diretórios que precisam continuar a pertencer ao grupo.

Finalmente, para excluir o grupo `web-developer`, execute o seguinte:

```
# groupdel web-developer
```

Não é possível excluir um grupo caso se trate do grupo principal de uma conta de usuário. Portanto, é preciso remover o usuário antes de remover o grupo. Quanto aos usuários, se você excluir um grupo, os arquivos pertencentes a esse grupo permanecerão em seu sistema de arquivos e não serão excluídos ou atribuídos a outro grupo.

O diretório de esqueleto

Quando você adiciona uma nova conta de usuário e cria seu diretório inicial, o diretório inicial recém-criado é preenchido com arquivos e pastas copiados do diretório de esqueleto (por padrão `/etc/skel`). A ideia é simples: um administrador de sistema deseja adicionar novos usuários que tenham os mesmos arquivos e diretórios em sua pasta pessoal. Portanto, caso queira personalizar os arquivos e pastas criados automaticamente no diretório inicial das novas contas de usuário, você deve adicionar esses novos arquivos e pastas ao diretório de esqueleto.

TIP Se quiser listar todos os arquivos e diretórios no diretório de esqueleto, use o comando `ls -al`.

O arquivo `/etc/login.defs`

No Linux, o arquivo `/etc/login.defs` especifica os parâmetros de configuração que controlam a criação de usuários e grupos. Além disso, os comandos mostrados nas seções anteriores utilizam os valores padrão deste arquivo.

As diretivas mais importantes são:

`UID_MIN` e `UID_MAX`

O intervalo de IDs de usuário que podem ser atribuídos a novos usuários comuns.

GID_MIN e GID_MAX

O intervalo de IDs de grupo que podem ser atribuídos a novos grupos comuns.

CREATE_HOME

Especifica se um diretório pessoal deve ser criado por padrão para novos usuários.

USERGROUPS_ENAB

Especifica se o sistema deve, por padrão, criar um novo grupo para cada nova conta de usuário com o mesmo nome do usuário, e se, ao deletar a conta do usuário, o grupo primário do usuário também deve ser removido, caso não contenha mais membros.

MAIL_DIR

O diretório de spool de email.

PASS_MAX_DAYS

O número máximo de dias que uma senha pode ser usada.

PASS_MIN_DAYS

O número mínimo de dias permitido entre mudanças de senha.

PASS_MIN_LEN

O comprimento mínimo aceitável da senha.

PASS_WARN_AGE

O número de dias de aviso antes que uma senha expire.

TIP Ao gerenciar usuários e grupos, sempre verifique este arquivo para visualizar e, eventualmente, alterar o comportamento padrão do sistema, se necessário.

O comando passwd

Este comando é usado principalmente para alterar a senha de um usuário. Conforme descrito anteriormente, qualquer usuário pode alterar sua própria senha, mas apenas o root pode alterar a senha de *qualquer* usuário. Isso acontece porque o comando `passwd` tem o bit SUID definido (um s no lugar do sinalizador executável para o proprietário), o que significa que ele é executado com os privilégios do proprietário do arquivo (portanto, root).

```
# ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 42096 mag 17 2015 /usr/bin/passwd
```

Dependendo das opções utilizadas com `passwd`, é possível controlar aspectos específicos do envelhecimento da senha:

-d

Apaga a senha de uma conta de usuário (desabilitando o usuário).

-e

Força a conta de usuário a alterar a senha.

-i

Define o número de dias de inatividade após a expiração de uma senha, durante os quais o usuário deve atualizar a senha (caso contrário, a conta será desabilitada).

-l

Bloqueia a conta de usuário (a senha criptografada é prefixada com um ponto de exclamação no arquivo `/etc/shadow`).

-n

Define o tempo de vida mínimo da senha.

-s

Exibe informações sobre o status da senha de uma conta de usuário específica.

-u

Desbloqueia a conta do usuário (o ponto de exclamação é removido do campo de senha no arquivo `/etc/shadow`).

-x

Define o tempo de vida máximo da senha.

-w

Define o número de dias de aviso antes que a senha expire, durante os quais o usuário é advertido de que a senha deve ser alterada.

NOTE

Os grupos também podem ter uma senha, que pode ser definida usando o comando `gpasswd`. Os usuários que não são membros de um grupo mas conhecem a senha podem ingressar nele temporariamente usando o comando `newgrp`. `gpasswd` também é usado para adicionar e remover usuários de um grupo e para definir a lista de administradores e membros comuns do grupo.

O comando chage

Este comando, que significa “alterar idade” (change age), é usado para alterar as informações de validade da senha de um usuário. O comando chage é restrito ao root, exceto com a opção `-l`, que pode ser executada por usuários comuns para listar informações sobre a expiração de senha de suas próprias contas.

As outras opções que se aplicam ao comando chage são:

-d

Define a última alteração de senha para uma conta de usuário.

-E

Define a data de expiração para uma conta de usuário.

-I

Define o número de dias de inatividade após a expiração de uma senha, durante os quais o usuário deve atualizar a senha (caso contrário, a conta será desabilitada).

-m

Define o tempo de vida mínimo da senha para uma conta de usuário.

-M

Define o tempo de vida máximo da senha para uma conta de usuário.

-w

Define o número de dias de aviso antes que a senha expire, durante os quais o usuário é advertido de que a senha deve ser alterada.

Exercícios Guiados

1. Identifique a finalidade correspondente a cada um dos comandos a seguir:

usermod -L	
passwd -u	
chage -E	
groupdel	
useradd -s	
groupadd -g	
userdel -r	
usermod -l	
groupmod -n	
useradd -m	

2. Identifique o comando chage correspondente a cada um dos seguintes comandos passwd:

passwd -n	
passwd -x	
passwd -w	
passwd -i	
passwd -S	

3. Explique em detalhes a finalidade dos comandos da questão anterior:

4. Que comandos podemos usar para bloquear uma conta de usuário? E quais comandos para desbloqueá-la?

Exercícios Exploratórios

1. Usando o comando `groupadd`, crie os grupos `administrators` e `developers`. Suponha que você esteja logado como root.

2. Depois de criar os grupos, execute o seguinte comando: `useradd -G administrators,developers kevin`. Quais operações esse comando realiza? Suponha que `CREATE_HOME` e `USERGROUPS_ENAB` em `/etc/login.defs` estão definidos como `yes`.

3. Crie um novo grupo chamado `designers`, renomeie-o como `web-designers` e adicione esse novo grupo aos grupos secundários da conta de usuário `kevin`. Identifique todos os grupos a que `kevin` pertence e seus IDs.

4. Remova apenas o grupo `developers` dos grupos secundários de `kevin`.

5. Defina a senha para a conta de usuário `kevin`.

6. Usando o comando `chage`, primeiro verifique a data de expiração da conta de usuário `kevin` e depois altere-a para 31 de dezembro de 2022. Que outro comando pode ser usado para alterar a data de expiração de uma conta de usuário?

7. Adicione uma nova conta de usuário chamada `emma` com UID 1050 e defina `administrators` como seu grupo principal e `developers` e `web-designers` como seus grupos secundários.

8. Mude o shell de login de `emma` para `/bin/sh`.

9. Remova as contas de usuário `emma` e `kevin` e os grupos `administrators`, `developers` e `web-designers`.

Resumo

Nesta lição, você aprendeu:

- Os fundamentos do gerenciamento de usuários e grupos no Linux.
- Como adicionar, modificar e remover contas de usuário.
- Como adicionar, modificar e remover contas de grupo.
- Manter o diretório de esqueleto.
- Editar o arquivo que controla a criação de usuários e grupos.
- Alterar as senhas das contas de usuário.
- Alterar as informações de validade da senha das contas de usuário.

Os seguintes arquivos e comandos foram discutidos nesta lição:

useradd

Cria uma nova conta de usuário.

usermod

Modifica uma conta de usuário.

userdel

Exclui uma conta de usuário.

groupadd

Cria uma nova conta de grupo.

groupmod

Modifica uma conta de grupo.

groupdel

Exclui uma conta de grupo.

passwd

Altera a senha das contas de usuário e controla todos os aspectos do envelhecimento da senha.

chage

Altera as informações de expiração da senha do usuário.

/etc/skel

A localização padrão do diretório de esqueleto.

/etc/login.defs

O arquivo que controla a criação de usuários e grupos e fornece valores padrão para diversos parâmetros de contas de usuário.

Respostas aos Exercícios Guiados

1. Identifique a finalidade correspondente a cada um dos comandos a seguir:

<code>usermod -L</code>	Bloquear a conta de usuário
<code>passwd -u</code>	Desbloquear a conta de usuário
<code>chage -E</code>	Definir a data de expiração da conta de usuário
<code>groupdel</code>	Excluir o grupo
<code>useradd -s</code>	Criar uma nova conta de usuário com um shell de login específico
<code>groupadd -g</code>	Criar um novo grupo com um GID específico
<code>userdel -r</code>	Remover a conta de usuário e todos os arquivos em seu diretório inicial, o próprio diretório inicial e o spool de email do usuário
<code>usermod -l</code>	Alterar o nome de login da conta de usuário
<code>groupmod -n</code>	Alterar o nome do grupo
<code>useradd -m</code>	Criar uma nova conta de usuário e seu diretório inicial

2. Identifique o comando `chage` correspondente a cada um dos seguintes comandos `passwd`:

<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. Explique em detalhes a finalidade dos comandos da questão anterior:

No Linux, podemos usar o comando `passwd -n` (ou `chage -m`) para definir o número mínimo de dias entre as mudanças de senha, o comando `passwd -x` (ou `chage -M`) para definir o número máximo de dias durante os quais uma senha é válida, o comando `passwd -w` (ou `chage -W`) para definir o número de dias de aviso antes que a senha expire, o comando `passwd -i` (ou `chage -I`) para definir o número de dias de inatividade durante os quais o usuário deve

alterar a senha e o comando `passwd -S` (ou `chage -l`) para exibir informações breves sobre a senha da conta de usuário.

- Que comandos podemos usar para bloquear uma conta de usuário? E quais comandos para desbloqueá-la?

Se quisermos bloquear uma conta de usuário, podemos usar um destes comandos: `usermod -L`, `usermod --lock` e `passwd -l`. Já para desbloqueá-la, os comandos seriam `usermod -U`, `usermod --unlock` e `passwd -u`.

Respostas aos Exercícios Exploratórios

1. Usando o comando `groupadd`, crie os grupos `administrators` e `developers`. Suponha que você esteja logado como root.

```
# groupadd administrators
# groupadd developers
```

2. Depois de criar os grupos, execute o seguinte comando: `useradd -G administrators,developers kevin`. Quais operações esse comando realiza? Suponha que `CREATE_HOME` e `USERGROUPS_ENAB` em `/etc/login.defs` estão definidos como `yes`.

O comando adiciona um novo usuário, chamado `kevin`, à lista de usuários do sistema, cria seu diretório inicial (`CREATE_HOME` é definido como sim e, portanto, a opção `-m` pode ser omitida) e cria um grupo, chamado `kevin`, como grupo principal dessa conta de usuário (`USERGROUPS_ENAB` é definido como sim). Finalmente, os arquivos e pastas contidos em um diretório de esqueleto são copiados para o diretório inicial de `kevin`.

3. Crie um novo grupo chamado `designers`, renomeie-o como `web-designers` e adicione esse novo grupo aos grupos secundários da conta de usuário `kevin`. Identifique todos os grupos a que `kevin` pertence e seus IDs.

```
# 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)
```

4. Remova apenas o grupo `developers` dos grupos secundários de `kevin`.

```
# usermod -G administrators,web-designers kevin
# id kevin
uid=1010(kevin) gid=1030(kevin) groups=1030(kevin),1028(administrators),1031(web-
designers)
```

O comando `usermod` não inclui a opção de remover apenas um grupo; portanto, você precisa especificar todos os grupos secundários aos quais o usuário pertence.

5. Defina a senha para a conta de usuário `kevin`.

```
# passwd kevin
Changing password for user kevin.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

6. Usando o comando `chage`, primeiro verifique a data de expiração da conta de usuário `kevin` e depois altere-a para 31 de dezembro de 2022. Que outro comando pode ser usado para alterar a data de expiração de uma conta de usuário?

```
# 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
```

O comando `usermod` com a opção `-e` equivale a `chage -E`.

7. Adicione uma nova conta de usuário chamada `emma` com UID 1050 e defina `administrators` como seu grupo principal e `developers` e `web-designers` como seus grupos secundários.

```
# 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. Mude o shell de login de `emma` para `/bin/sh`.

```
# usermod -s /bin/sh emma
```

9. Remova as contas de usuário `emma` e `kevin` e os grupos `administrators`, `developers` e `web-designers`.

```
# userdel -r emma
# userdel -r kevin
# groupdel administrators
# groupdel developers
# groupdel web-designers
```



107.1 Lição 2

Certificação:	LPIC-1
Versão:	5.0
Tópico:	107 Tarefas administrativas
Objetivo:	107.1 Gerenciar contas de usuários e grupos e arquivos de sistema relacionados
Lição:	2 de 2

Introdução

As ferramentas de linha de comando discutidas na lição anterior e os aplicativos gráficos (fornecidos pelas distribuições) que executam as mesmas tarefas atualizam uma série de arquivos nos quais são armazenadas informações sobre os usuários e grupos.

Esses arquivos ficam no diretório `/etc/` e são:

/etc/passwd

Um arquivo de sete campos delimitados por dois pontos contendo informações básicas sobre os usuários.

/etc/group

Um arquivo de quatro campos delimitados por dois pontos contendo informações básicas sobre os grupos.

/etc/shadow

Um arquivo de nove campos delimitados por dois pontos contendo senhas de usuário

criptografadas.

/etc/gshadow

Um arquivo de quatro campos delimitados por dois pontos contendo senhas de grupo criptografadas.

Embora esses quatro arquivos estejam em texto simples, eles não devem jamais ser editados diretamente, mas sim por meio das ferramentas fornecidas pela distribuição que se está usando.

/etc/passwd

Este é um arquivo legível por todos contendo uma lista de usuários em linhas separadas. Cada linha consiste em sete campos delimitados por dois pontos:

Nome de usuário

O nome usado quando o usuário se loga no sistema.

Senha

A senha criptografada (ou um x no caso de senhas shadow).

ID de usuário (UID)

O número de identificação atribuído ao usuário no sistema.

ID de grupo (GID)

O número do grupo principal do usuário no sistema.

GECOS

Um campo de comentário opcional, usado para adicionar informações extras sobre o usuário (como o nome completo). O campo pode conter várias entradas separadas por vírgulas.

Diretório inicial

O caminho absoluto do diretório inicial do usuário.

Shell

O caminho absoluto do programa que é iniciado automaticamente quando o usuário efetua login no sistema (geralmente um shell interativo como /bin/bash).

/etc/group

Este é um arquivo legível por todos contendo uma lista de grupos em linhas separadas. Cada linha consiste em quatro campos delimitados por dois pontos:

Nome do grupo

O nome do grupo.

Senha do grupo

A senha criptografada do grupo (ou um x se forem usadas senhas shadow).

ID do grupo (GID)

O número de identificação atribuído ao grupo no sistema.

Lista de membros

Uma lista delimitada por vírgulas de usuários pertencentes ao grupo, exceto aqueles para os quais este é o grupo principal.

/etc/shadow

Este é um arquivo que pode ser lido apenas pelo root e por usuários com privilégios de root contendo senhas de usuário criptografadas em linhas separadas. Cada linha consiste em nove campos delimitados por dois pontos:

Nome de usuário

O nome usado quando o usuário se loga no sistema.

Senha criptografada

A senha criptografada do usuário (se o valor começar com !, a conta está bloqueada).

Data da última mudança de senha

A data da última alteração de senha, em número de dias desde 01/01/1970 (o valor 0 significa que o usuário deve alterar a senha no próximo login).

Idade mínima da senha

O número mínimo de dias que devem decorrer após uma alteração de senha para que o usuário tenha permissão de alterar a senha novamente.

Idade máxima da senha

O número máximo de dias que devem se passar antes que uma alteração de senha seja necessária.

Período de aviso de senha

O número de dias até a expiração da senha, durante os quais o usuário é avisado de que a senha deve ser alterada.

Período de inatividade da senha

O número de dias após a expiração de uma senha, durante os quais o usuário deve atualizá-la. Após esse período, se o usuário não alterar a senha, a conta é desativada.

Data de expiração da conta

A data, em número de dias desde 01/01/1970, na qual a conta do usuário será desativada. Um campo vazio indica que a conta do usuário nunca expirará.

Um campo reservado

Um campo reservado para uso futuro.

/etc/gshadow

Este é um arquivo legível apenas pelo root e por usuários com privilégios de root que contém senhas criptografadas para grupos em linhas separadas. Cada linha consiste em quatro campos delimitados por dois pontos:

Nome do grupo

O nome do grupo.

Senha criptografada

A senha criptografada do grupo (é usada quando um usuário que não é membro do grupo deseja ingressar no grupo usando o comando `newgrp` -- se a senha começar com `!`, ninguém tem permissão de acessar o grupo com `newgrp`).

Administradores do grupo

Uma lista dos administradores do grupo delimitada por vírgulas (eles podem alterar a senha do grupo, bem como adicionar ou remover membros do grupo com o comando `gpasswd`).

Membros do grupo

Uma lista dos membros do grupo delimitada por vírgulas.

Como filtrar os bancos de dados de senha e grupo

É frequentemente necessário rever as informações sobre usuários e grupos armazenadas nesses quatro arquivos e pesquisar por registros específicos. Para realizar esta tarefa, usamos o comando `grep` ou, alternativamente, concatenamos `cat` e `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
```

Outra maneira de acessar esses bancos de dados é usar o comando `getent`. Em geral, esse comando exibe entradas de bancos de dados suportados pelas bibliotecas *Name Service Switch* (NSS) e requer o nome do banco de dados e uma chave de pesquisa. Se nenhum argumento-chave for fornecido, todas as entradas do banco de dados especificado serão exibidas (a menos que o banco de dados não suporte enumeração). Caso contrário, se um ou mais argumentos-chave forem fornecidos, o banco de dados será filtrado de acordo.

```
# getent passwd emma
emma:x:1020:1020:User Emma:/home/emma:/bin/bash
# getent group db-admin
db-admin:x:1050:grace,frank
```

O comando `getent` não requer autoridade de root; você só precisa ter a permissão para ler o banco de dados do qual deseja recuperar os registros.

NOTE

Lembre-se de que o `getent` só pode acessar os bancos de dados configurados no arquivo `/etc/nsswitch.conf`.

Exercícios Guiados

- Analise esta saída e responda às questões a seguir:

```
# 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.jC1T06ljsdczvklPkpi/amgp.zyfAN680zrLLp2avvpd
KA0llpssdfcPppOp:18015:0:99999:7:::
mail:*:18015:0:99999:7:::
catherine:$6$ABCD25jlld14hpPthEFGnnssEWw1234yioMpliABCdef1f3478kAfhhAfgbAMjY1/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
```

- Qual o identificador de usuário (UID) e identificador de grupo (GID) de root e catherine?

- Qual o nome do grupo principal de kevin? Existem outros membros neste grupo?

- Qual shell está configurado para mail? O que isso significa?

- Quem são os membros do grupo app-developer? Quais deles são administradores e quais são membros comuns do grupo?

- Qual o tempo de vida mínimo da senha de **catherine**? E qual o tempo de vida máximo da senha?

- Qual o período de inatividade da senha para **kevin**?

2. Por convenção, quais IDs são atribuídos às contas do sistema e quais aos usuários comuns?

3. Como você descobre se uma conta de usuário, anteriormente capaz de acessar o sistema, agora está bloqueada? Suponha que seu sistema use senhas shadow.

Exercícios Exploratórios

1. Crie uma conta de usuário chamada `christian` usando o comando `useradd -m` e identifique sua ID de usuário (UID), ID de grupo (GID) e shell.

2. Identifique o nome do grupo principal de `christian`. O que podemos deduzir?

3. Usando o comando `getent`, reveja as informações de validade da senha para a conta de usuário `christian`.

4. Adicione o grupo `editor` aos grupos secundários de `christian`. Suponha que este grupo já contenha `emma`, `dave` e `frank` como membros comuns. Como você pode conferir se não há administradores para este grupo?

5. Execute o comando `ls -l /etc/passwd /etc/group /etc/shadow /etc/gshadow` e descreva a saída em termos de permissões de arquivos. Quais desses quatro arquivos estão em shadow por razões de segurança? Pressuponha que seu sistema usa senhas shadow.

Resumo

Nesta lição, você aprendeu:

- A localização dos arquivos que armazenam informações sobre usuários e grupos.
- Gerenciar informações de usuários e grupos armazenadas em bancos de dados de senhas e grupos.
- Recuperar informações de bancos de dados de senha e grupo.

Os seguintes arquivos e comandos foram discutidos nesta lição:

/etc/passwd

O arquivo que contém informações básicas sobre os usuários.

/etc/group

O arquivo que contém informações básicas sobre grupos.

/etc/shadow

O arquivo que contém senhas de usuário criptografadas.

/etc/gshadow

O arquivo que contém as senhas de grupo criptografadas.

getent

Filtra os bancos de dados de senha e grupo.

Respostas aos Exercícios Guiados

- Analise esta saída e responda às questões a seguir:

```
# 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.jC1T06ljsdczvklPkpi/amgp.zyfAN680zrLLp2avvpd
KA0llpssdfcPppOp:18015:0:99999:7:::
mail:*:18015:0:99999:7:::
catherine:$6$ABCD25jlld14hpPthEFGnnssEWw1234yioMpliABCdef1f3478kAfhhAfgbAMjY1/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
```

- Qual o identificador de usuário (UID) e identificador de grupo (GID) de `root` e `catherine`?

O UID e GID de `root` são 0 e 0, ao passo que o UID e GID de `catherine` são 1030 e 1025.

- Qual o nome do grupo principal de `kevin`? Existem outros membros neste grupo?

O nome do grupo é `db-admin`. `emma` e `grace` também estão nesse grupo.

- Qual shell está configurado para `mail`? O que isso significa?

`mail` é uma conta de usuário do sistema e seu shell é `/sbin/nologin`. Na verdade, as contas de usuário do sistema como `mail`, `ftp`, `news` e `daemon` são usadas para realizar tarefas administrativas e, portanto, é necessário evitar que essas contas façam login normalmente. É por essa razão que o shell costuma ser configurado para `/sbin/nologin` ou `/bin/false`.

- Quem são os membros do grupo `app-developer`? Quais deles são administradores e quais são membros comuns do grupo?

Os membros são `catherine`, `dave` e `christian`, e eles são todos membros comuns.

- Qual o tempo de vida mínimo da senha de `catherine`? E qual o tempo de vida máximo da senha?

A vida útil mínima da senha é de 20 dias e a vida útil máxima da senha é 90 dias.

- Qual o período de inatividade da senha para `kevin`?

O período de inatividade da senha é de 2 dias. Durante este período, `kevin` deve atualizar a senha, caso contrário a conta será desativada.

2. Por convenção, quais IDs são atribuídos às contas do sistema e quais aos usuários comuns?

As contas do sistema geralmente têm UIDs menores que 100 ou entre 500 e 1000, enquanto os usuários comuns têm UIDs começando em 1000 (embora alguns sistemas legados comecem a numerar em 500). O usuário `root` tem UID 0. Os valores `UID_MIN` e `UID_MAX` em `/etc/login.defs` definem a faixa de UIDs usados para a criação de usuários comuns. Do ponto de vista do LPI Linux Essentials e do LPIC-1, as contas do sistema têm UIDs menores que 1000 e os usuários comuns têm UIDs maiores que 1000.

3. Como você descobre se uma conta de usuário, anteriormente capaz de acessar o sistema, agora está bloqueada? Suponha que seu sistema use senhas shadow.

Quando se usam senhas shadow, o segundo campo em `/etc/passwd` contém o caractere `x` de cada conta de usuário, pois as senhas de usuário criptografadas são armazenadas em `/etc/shadow`. Em particular, a senha criptografada de uma conta de usuário é armazenada no segundo campo deste arquivo e, se ela iniciar com um ponto de exclamação, a conta está bloqueada.

Respostas aos Exercícios Exploratórios

- Crie uma conta de usuário chamada `christian` usando o comando `useradd -m` e identifique sua ID de usuário (UID), ID de grupo (GID) e shell.

```
# useradd -m christian
# cat /etc/passwd | grep christian
christian:x:1050:1060::/home/christian:/bin/bash
```

O UID e GID de `christian` são 1050 e 1060 respectivamente (o terceiro e o quarto campos em `/etc/passwd`). `/bin/bash` é o shell definido para esta conta de usuário (o sétimo campo em `/etc/passwd`).

- Identifique o nome do grupo principal de `christian`. O que podemos deduzir?

```
# cat /etc/group | grep 1060
christian:x:1060:
```

O nome do grupo principal de `christian` é `christian` (o primeiro campo em `/etc/group`). Portanto, `USERGROUPS_ENAB` em `/etc/login.defs` é definido como sim, de modo que `useradd` cria, por padrão, um grupo com o mesmo nome da conta de usuário.

- Usando o comando `getent`, reveja as informações de validade da senha para a conta de usuário `christian`.

```
# getent shadow christian
christian:!:18015:0:99999:7:::
```

A conta de usuário `christian` não tem a senha definida e agora está bloqueada (o segundo campo em `/etc/shadow` contém um ponto de exclamação). Não há idade mínima e máxima de senha para esta conta de usuário (o quarto e o quinto campos em `/etc/shadow` estão definidos como 0 e 99999 dias), ao passo que o período de aviso de senha está configurado para 7 dias (o sexto campo em `/etc/shadow`). Finalmente, não há período de inatividade (o sétimo campo em `/etc/shadow`) e a conta jamais expira (o oitavo campo em `/etc/shadow`).

- Adicione o grupo `editor` aos grupos secundários de `christian`. Suponha que este grupo já contenha `emma`, `dave` e `frank` como membros comuns. Como você pode conferir se não há administradores para este grupo?

```
# 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
```

O terceiro e quarto campos em `/etc/gshadow` contêm administradores e membros comuns do grupo especificado. Portanto, como o terceiro campo de `editor` está vazio, este grupo não tem administradores (`emma, dave, frank` e `christian` são todos membros comuns).

- Execute o comando `ls -l /etc/passwd /etc/group /etc/shadow /etc/gshadow` e descreva a saída em termos de permissões de arquivos. Quais desses quatro arquivos estão em shadow por razões de segurança? Pressuponha que seu sistema usa senhas shadow.

```
# 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
```

Os arquivos `/etc/passwd` e `/etc/group` são legíveis por todos e estão na sombra por motivos de segurança. Quando senhas shadow são usadas, vemos um `x` no segundo campo desses arquivos, pois as senhas criptografadas de usuários e grupos são armazenadas em `/etc/shadow` e `/etc/gshadow`, que são legíveis somente por root e, em meu sistema, até mesmo pelos membros do grupo `shadow`.



Linux
Professional
Institute

107.2 Automatizar e agendar tarefas administrativas de sistema

Referência ao LPI objectivo

LPIC-1 version 5.0, Exam 102, Objective 107.2

Peso

4

Áreas chave de conhecimento

- Gerenciar tarefas usando cron e at.
- Configurar o acesso dos usuários a serviços cron e at.
- Entender as unidades temporizadoras (timers) do systemd.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- /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`



**Linux
Professional
Institute**

107.2 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	107 Tarefas administrativas
Objetivo:	107.2 Automatizar tarefas de administração do sistema agendando trabalhos
Lição:	1 de 2

Introdução

Uma das funções mais importantes de um bom administrador de sistema é agendar os jobs (trabalhos) que precisam ser executados regularmente. Por exemplo, um administrador pode criar e automatizar jobs para fazer backups, atualizações do sistema e muitas outras atividades repetitivas. Para isso, usamos o recurso `cron`, que serve para automatizar o agendamento de tarefas periódicas.

Como agendar jobs com o cron

No Linux, `cron` é um daemon que roda continuamente e desperta a cada minuto para verificar uma série de tabelas em busca de tarefas a executar. Essas tabelas são chamadas de *crontabs* e contêm os chamados *cron jobs* (trabalhos cron). O Cron é adequado para servidores e sistemas que estão constantemente ligados, pois cada cron job é executado somente se o sistema estiver rodando no horário programado. Ele pode ser usado por usuários comuns, cada um com seu próprio *crontab*, bem como pelo usuário root, que gerencia os crontabs do sistema.

NOTE No Linux, também existe o recurso `anacron`, adequado para sistemas que podem

ser desligados (como computadores de mesa ou laptops). Ele só pode ser usado pelo root. Se a máquina estiver desligada quando os trabalhos de anacron tiverem de ser executados, isso ocorrerá na próxima vez em que se ligar a máquina. O anacron está fora do escopo da certificação LPIC-1.

Crontabs de usuário

Os *crontabs de usuário* são arquivos de texto que gerenciam o agendamento de trabalhos cron definidos pelo usuário. Eles sempre têm o nome da conta de usuário que os criou, mas a localização desses arquivos depende da distribuição usada (geralmente é um subdiretório de `/var/spool/cron`).

Cada linha em um crontab de usuário contém seis campos separados por um espaço:

- O minuto da hora (0-59).
- A hora do dia (0-23).
- O dia do mês (1-31).
- O mês do ano (1-12).
- O dia da semana (0-7 com Domingo=0 ou Domingo=7).
- O comando a executar.

Para o mês do ano e o dia da semana, podemos usar as três primeiras letras do nome em vez do número correspondente.

Os primeiros cinco campos indicam quando executar o comando especificado no sexto campo e podem conter um ou mais valores. Em particular, é possível especificar vários valores usando:

* (asterisco)

Refere-se a qualquer valor.

, (vírgula)

Especifica uma lista de valores possíveis.

- (hífen)

Especifica um intervalo de valores possíveis.

/ (barra)

Especifica valores escalonados.

Muitas distribuições incluem o arquivo `/etc/crontab`, que pode ser usado como referência para a disposição dos elementos de um arquivo cron. Eis um exemplo de arquivo `/etc/crontab` de uma instalação de 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
```

Crontabs de sistema

Os *crontabs de sistema* são arquivos de texto que gerenciam a programação de cron jobs do sistema e só podem ser editados pelo usuário root. `/etc/crontab` e todos os arquivos no diretório `/etc/cron.d` são crontabs de sistema.

A maioria das distribuições inclui também os diretórios `/etc/cron.hourly` (de hora em hora), `/etc/cron.daily` (diariamente), `/etc/cron.weekly` (semanalmente) e `/etc/cron.monthly` (mensalmente), que contêm scripts que devem ser executados com a frequência adequada. Por exemplo, se você quiser executar um script diariamente, pode colocá-lo em `/etc/cron.daily`.

WARNING

Algumas distribuições usam `/etc/cron.d/hourly`, `/etc/cron.d/daily`, `/etc/cron.d/weekly` e `/etc/cron.d/monthly`. Lembre-se de sempre conferir os diretórios corretos nos quais colocar os scripts que o cron deve executar.

A sintaxe dos crontabs de sistema é semelhante à dos crontabs de usuário, porém ela requer um campo adicional obrigatório que especifica qual usuário executará o cron job. Portanto, cada linha em um crontab de sistema contém sete campos separados por um espaço:

- O minuto da hora (0-59).
- A hora do dia (0-23).
- O dia do mês (1-31).
- O mês do ano (1-12).

- O dia da semana (0-7 com Domingo=0 ou Domingo=7).
- O nome da conta de usuário a ser usada ao executar o comando.
- O comando a executar.

Quanto aos crontabs do usuário, podemos especificar mais de um valor nos campos de tempo usando os operadores *, , , - e /. Também é possível indicar o mês do ano e o dia da semana com as três primeiras letras do nome em vez do número correspondente.

Especificações de tempo particulares

Ao editar os arquivos crontab, podemos usar atalhos especiais nas primeiras cinco colunas em vez das especificações de tempo:

@reboot

Roda a tarefa especificada uma vez após a reinicialização.

@hourly

Roda a tarefa especificada uma vez por hora no início da hora.

@daily (or @midnight)

Roda a tarefa especificada uma vez por dia à meia-noite.

@weekly

Roda a tarefa especificada uma vez por semana, à meia-noite de domingo.

@monthly

Roda a tarefa especificada uma vez por mês, à meia-noite do primeiro dia do mês.

@yearly (or @annually)

Roda a tarefa especificada uma vez por ano, à meia-noite de 1º de janeiro.

Variáveis no crontab

Dentro de um arquivo crontab, pode haver atribuições de variáveis definidas antes que as tarefas agendadas sejam declaradas. As variáveis de ambiente comumente definidas são:

HOME

O diretório no qual o cron invoca os comandos (por padrão, o diretório inicial do usuário).

MAILTO

O nome do usuário ou o endereço para o qual a saída e o erro padrão são enviados (por padrão, o proprietário do crontab). Diversos valores separados por vírgulas também são permitidos, e um valor vazio indica que nenhum email deve ser enviado.

PATH

O caminho no qual os comandos podem ser encontrados.

SHELL

O shell a ser usado (por padrão `/bin/sh`).

Criando cron jobs de usuário

O comando `crontab` é usado para manter arquivos crontab para usuários individuais. Em particular, o comando `crontab -e` serve para editar seu próprio arquivo crontab ou para criar um, caso ele ainda não exista.

```
$ 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]:
```

Por padrão, o comando `crontab` abre o editor especificado pelas variáveis de ambiente `VISUAL` ou `EDITOR` para que você possa começar a editar seu arquivo crontab com o editor de sua preferência. Algumas distribuições, como mostrado no exemplo acima, permitem escolher o editor em uma lista quando `crontab` é executado pela primeira vez.

Se você deseja executar o script `foo.sh`, localizado em seu diretório inicial, todos os dias às 10:00, adicione a seguinte linha ao seu arquivo crontab:

```
0 10 * * * /home/frank/foo.sh
```

Considere os seguintes exemplos de entradas de crontab:

```
0,15,30,45 08 * * 2 /home/frank/bar.sh
30 20 1-15 1,6 1-5 /home/frank/foobar.sh
```

Na primeira linha, o script `bar.sh` é executado todas as terças-feiras às 08:00, às 08:15, às 08:30 e às 08:45. Na segunda linha, o script `foobar.sh` é executado às 20h30 de segunda a sexta-feira durante os primeiros quinze dias de janeiro e junho.

WARNING Embora os arquivos crontab possam ser editados manualmente, é sempre recomendável usar o comando `crontab`. As permissões nos arquivos crontab geralmente possibilitam editá-los por meio do comando `crontab`.

Além da opção `-e` mencionada acima, o comando `crontab` inclui outras opções úteis:

-l

Exibe o crontab atual na saída padrão.

-r

Remove o crontab atual.

-u

Especifica o nome do usuário cujo crontab precisa ser modificado. Esta opção requer privilégios de root e permite que o usuário root edite os arquivos crontab do usuário.

Criando cron jobs do sistema

Ao contrário dos crontabs de usuário, os crontabs de sistema são atualizados com um editor; portanto, não é preciso executar o comando `crontab` para editar `/etc/crontab` e os arquivos em `/etc/cron.d`. Lembre-se de que, ao editar crontabs do sistema, é necessário especificar a conta que será usada para executar o cron job (geralmente o usuário root).

Por exemplo, para executar o script `barfoo.sh` localizado no diretório `/root` todos os dias à 01:30, abra `/etc/crontab` com seu editor preferido e adicione a seguinte linha:

```
30 01 * * * root /root/barfoo.sh >>/root/output.log 2>>/root/error.log
```

No exemplo acima, a saída do job é anexada a `/root/output.log`, enquanto os erros são anexados a `/root/error.log`.

WARNING

Exceto nos casos em que a saída é redirecionada para um arquivo, como no exemplo acima (ou se a variável `MAILTO` estiver definida como um valor em

branco), toda a saída de um trabalho cron será enviada ao usuário via email. Uma prática comum é redirecionar a saída padrão para `/dev/null` (ou para um arquivo, para revisão posterior, se necessário) e não redirecionar o erro padrão. Desta forma, o usuário é notificado imediatamente por email sobre eventuais erros.

Configurando o acesso ao agendamento de trabalhos

No Linux, os arquivos `/etc/cron.allow` e `/etc/cron.deny` são usados para definir as restrições do `crontab`. Em particular, eles servem para permitir ou proibir o agendamento de trabalhos cron para diferentes usuários. Se `/etc/cron.allow` existir, apenas usuários não root listados nele podem agendar trabalhos cron usando o comando `crontab`. Se `/etc/cron.allow` não existir, mas `/etc/cron.deny` sim, apenas usuários não root listados neste arquivo não podem agendar trabalhos cron usando o comando `crontab` (neste caso, um `/etc/cron.deny` vazio significa que todos os usuários têm permissão para agendar trabalhos cron com o `crontab`). Se nenhum desses arquivos existir, o acesso do usuário ao agendamento de trabalhos cron dependerá da distribuição usada.

NOTE

Os arquivos `/etc/cron.allow` e `/etc/cron.deny` contêm uma lista de nomes de usuários separados por linhas.

Uma alternativa ao cron

Usando o `systemd` como gerenciador de sistema e serviços, podemos definir *temporizadores* como alternativa ao `cron` para agendar tarefas. Temporizadores são arquivos da unidade `systemd` identificados pelo sufixo `.timer`. Para cada um deles, deve haver um arquivo de unidade correspondente descrevendo a unidade a ser ativada quando o temporizador terminar. Por padrão, um `timer` ativa um serviço com o mesmo nome, exceto pelo sufixo.

Um temporizador inclui uma seção `[Timer]` que especifica quando os jobs agendados devem ser executados. Especificamente, podemos usar a opção `OnCalendar=` para definir *temporizadores em tempo real*, que funcionam da mesma maneira que os cron jobs (baseiam-se na expressão de eventos de calendário). A opção `OnCalendar=` requer a seguinte sintaxe:

```
DayOfWeek Year-Month-Day Hour:Minute:Second
```

sendo `DayOfWeek` opcional. Os operadores `*`, `/` e `,` têm o mesmo significado dos que são usados para cron jobs. Usamos `..` entre dois valores para indicar um intervalo contíguo. Na especificação `DayOfWeek` (dia da semana), podemos usar as primeiras três letras do nome ou o nome completo.

NOTE Você também pode definir *temporizadores monotônicos*, que são ativados após algum tempo decorrido de um ponto de início específico (por exemplo, quando a máquina foi inicializada ou quando o próprio temporizador foi ativado).

Por exemplo, para rodar o serviço `/etc/systemd/system/foobar.service` às 05:30 da primeira segunda-feira do mês, adicionamos as seguintes linhas no arquivo de unidade `/etc/systemd/system/foobar.timer` correspondente:

```
[Unit]
Description=Run the foobar service

[Timer]
OnCalendar=Mon *-* 05:30:00
Persistent=true

[Install]
WantedBy=timers.target
```

Depois de criar o novo temporizador, você pode ativá-lo e iniciá-lo executando os seguintes comandos como root:

```
# systemctl enable foobar.timer
# systemctl start foobar.timer
```

Podemos alterar a frequência do trabalho agendado modificando o valor `OnCalendar` e, em seguida, digitando o comando `systemctl daemon-reload`.

Finalmente, se você quiser ver a lista de temporizadores ativos ordenados pelo momento em que terminam, use o comando `systemctl list-timers`. A opção `--all` exibe também as unidades de temporizador inativas.

NOTE Lembre-se de que os temporizadores são registrados no diário do `systemd` e você pode rever os registros das diferentes unidades usando o comando `journalctl`. Além disso, se estiver trabalhando como um usuário comum, será preciso usar a opção `--user` dos comandos `systemctl` e `journalctl`.

Em vez da forma normalizada mais longa mencionada acima, é possível usar algumas expressões especiais que descrevem frequências específicas para a execução de um job:

hourly

Roda a tarefa especificada uma vez por hora, no início da hora.

daily

Roda a tarefa especificada uma vez por dia à meia-noite.

weekly

Roda a tarefa especificada uma vez por semana, na meia-noite de segunda-feira.

monthly

Roda a tarefa especificada uma vez por mês, na meia-noite do primeiro dia do mês.

yearly

Roda a tarefa especificada uma vez por ano, na meia-noite de 1º de janeiro.

Consulte as páginas de manual para ver a lista completa de especificações de hora e data em `systemd.timer(5)`.

Exercícios Guiados

1. Para cada um dos seguintes atalhos de crontab, indique a especificação de tempo correspondente (ou seja, as cinco primeiras colunas de um arquivo crontab de usuário):

@hourly	
@daily	
@weekly	
@monthly	
@annually	

2. Para cada um dos seguintes atalhos de OnCalendar, indique a especificação de tempo correspondente (a forma normalizada mais longa):

hourly	
daily	
weekly	
monthly	
yearly	

3. Explique o significado das seguintes especificações de tempo encontradas em um arquivo 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. Explique o significado das seguintes especificações de tempo usadas na opção OnCalendar de um arquivo de temporizador:

--* 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	

Exercícios Exploratórios

1. Pressupondo que você esteja autorizado a agendar tarefas com o `cron` como um usuário comum, qual comando você usaria para criar seu próprio arquivo `crontab`?

2. Crie um job agendado simples que execute o comando `date` todas as sextas-feiras às 13h. Onde você poderia ver o resultado deste trabalho?

3. Crie outro job agendado que execute o script `foobar.sh` a cada minuto, redirecionando a saída para o arquivo `output.log` em seu diretório inicial de forma que apenas o erro padrão seja enviado a você por email.

4. Observe a entrada `crontab` do job agendado que acaba de criar. Por que não é necessário especificar o caminho absoluto do arquivo no qual a saída padrão é salva? E por que podemos usar o comando `./foobar.sh` para executar o script?

5. Edite a entrada `crontab` anterior removendo o redirecionamento de saída e desabilite o primeiro cron job que criou.

6. Como é possível enviar a saída e os erros do seu trabalho agendado para a conta de usuário `emma` via email? E como evitar o envio da saída padrão e erros por email?

7. Execute o comando `ls -l /usr/bin/crontab`. Qual bit especial está definido e qual é o seu significado?

Resumo

Nesta lição, você aprendeu:

- Usar `cron` para executar jobs em intervalos regulares.
- Gerenciar os cron jobs.
- Configurar o acesso de usuário ao agendamento de cron jobs.
- Entender o papel das unidades de temporizador do `systemd` como alternativa ao `cron`.

Os seguintes comandos e arquivos foram discutidos nesta lição:

crontab

Mantém os arquivos `crontab` de usuários individuais.

/etc/cron.allow e /etc/cron.deny

Arquivos particulares usados para definir as restrições do `crontab`.

/etc/crontab

Arquivo `crontab` do sistema.

/etc/cron.d

O diretório que contém os arquivos `crontab` do sistema.

systemctl

Controla o sistema e o gerenciador de serviços `systemd`. Em relação aos temporizadores, pode ser utilizado para habilitá-los e iniciá-los.

Respostas aos Exercícios Guiados

1. Para cada um dos seguintes atalhos de crontab, indique a especificação de tempo correspondente (ou seja, as cinco primeiras colunas de um arquivo crontab de usuário):

@hourly	0 * * * *
@daily	0 0 * * *
@weekly	0 0 * * 0
@monthly	0 0 1 * *
@annually	0 0 1 1 *

2. Para cada um dos seguintes atalhos de OnCalendar, indique a especificação de tempo correspondente (a forma normalizada mais longa):

hourly	*-*-* *:00:00
daily	*-*-* 00:00:00
weekly	Mon *-*-* 00:00:00
monthly	*-*-* 01 00:00:00
yearly	*-*-* 01-01 00:00:00

3. Explique o significado das seguintes especificações de tempo encontradas em um arquivo crontab:

30 13 * * 1-5	Às 13h30 todos os dias da semana, de segunda a sexta-feira
00 09-18 * * *	Todos os dias e todas as horas das 09h00 às 18h00
30 08 1 1 *	Às 08h30 do primeiro dia de janeiro
0,20,40 11 * * Sun	Todos os domingos às 11h, 11h20 e 11h40
00 09 10-20 1-3 *	Às 09h00 de 10 a 20 de janeiro, fevereiro e março
*/20 * * * *	A cada vinte minutos

4. Explique o significado das seguintes especificações de tempo usadas na opção OnCalendar de um arquivo de temporizador:

<code>*-*-* 08:30:00</code>	Todos os dias às 08:30
<code>Sat,Sun *-*-* 05:00:00</code>	Sábado e domingo às 05h00
<code>*-*-* 13:15,30,45:00</code>	Às 13h15, 13h30 e 13h45 do primeiro dia do mês
<code>Fri *-09..12-* 16:20:00</code>	Às 16h20 todas as sextas-feiras de setembro, outubro, novembro e dezembro
<code>Mon,Tue *-*-* 08:30:00</code>	Às 08h30 do primeiro ou décimo quinto dia de cada mês, apenas se o dia for segunda ou terça-feira
<code>*-*-* *:00/05:00</code>	A cada cinco minutos

Respostas aos Exercícios Exploratórios

- Pressupondo que você esteja autorizado a agendar tarefas com o `cron` como um usuário comum, qual comando você usaria para criar seu próprio arquivo `crontab`?

```
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]:
```

- Crie um job agendado simples que execute o comando `date` todas as sextas-feiras às 13h. Onde você poderia ver o resultado deste trabalho?

```
00 13 * * 5 date
```

A saída é enviada ao usuário; para visualizá-la, use o comando `mail`.

- Crie outro job agendado que execute o script `foobar.sh` a cada minuto, redirecionando a saída para o arquivo `output.log` em seu diretório inicial de forma que apenas o erro padrão seja enviado a você por email.

```
*/1 * * * * ./foobar.sh >> output.log
```

- Observe a entrada `crontab` do job agendado que acaba de criar. Por que não é necessário especificar o caminho absoluto do arquivo no qual a saída padrão é salva? E por que podemos usar o comando `./foobar.sh` para executar o script?

`cron` invoca os comandos do diretório inicial do usuário, a menos que outro local seja especificado pela variável de ambiente `HOME` dentro do arquivo `crontab`. Por esta razão, podemos usar o caminho relativo do arquivo de saída e executar o script com `./foobar.sh`.

- Edite a entrada `crontab` anterior removendo o redirecionamento de saída e desabilite o primeiro cron job que criou.

```
#00 13 * * 5 date
*/1 * * * * ./foobar.sh
```

Para desabilitar um trabalho cron, podemos simplesmente comentar a linha correspondente dentro do arquivo `crontab`.

- Como é possível enviar a saída e os erros do seu trabalho agendado para a conta de usuário `emma` via email? E como evitar o envio da saída padrão e erros por email?

Para enviar a saída padrão e o erro para `emma`, definimos a variável de ambiente `MAILTO` em nosso arquivo `crontab` desta forma:

```
MAILTO="emma"
```

Para dizer ao `cron` que nenhum email deve ser enviado, atribuímos um valor vazio à variável de ambiente `MAILTO`.

```
MAILTO=""
```

- Execute o comando `ls -l /usr/bin/crontab`. Qual bit especial está definido e qual é o seu significado?

```
$ ls -l /usr/bin/crontab
-rwxr-sr-x 1 root crontab 25104 feb 10 2015 /usr/bin/crontab
```

O comando `crontab` tem o bit SGID definido (o caractere `s` ao invés do sinalizador executável para o grupo), o que significa que é executado com os privilégios do grupo (portanto, `crontab`). É por isso que usuários comuns podem editar seu arquivo `crontab` usando o comando `crontab`. Note que muitas distribuições definem as permissões de arquivo de tal forma que os arquivos `crontab` só podem ser editados por meio do comando `crontab`.



Linux
Professional
Institute

107.2 Lição 2

Certificação:	LPIC-1
Versão:	5.0
Tópico:	107 Tarefas administrativas
Objetivo:	107.2 Automatizar e agendar tarefas administrativas do sistema
Lição:	2 de 2

Introdução

Como vimos na lição anterior, podemos agendar tarefas regulares usando o cron ou os temporizadores do systemd, mas às vezes precisamos executar uma tarefa apenas uma vez em um momento específico do futuro. Para isso, existe outro utilitário poderoso: o comando at.

Agendamento de trabalhos com at

O comando at é usado para agendamento de tarefas únicas e requer apenas que se especifique quando o trabalho deve ser executado no futuro. Após inserir at na linha de comando seguido pela especificação de tempo, você entrará no prompt de at para definir os comandos a serem executados. Saia do prompt com a seqüência de teclas **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
```

O job `at` no exemplo acima simplesmente executa o comando `date` após cinco minutos. Como no caso do `cron`, a saída padrão e o erro são enviados por email. Observe que o daemon `atd` precisará estar rodando no sistema para ser possível usar o agendamento de tarefas `at`.

NOTE No Linux, o comando `batch` é semelhante a `at`, porém os jobs `batch` são executados apenas quando a carga do sistema está baixa o suficiente para permiti-lo.

As opções mais importantes do comando `at` são:

-c

Imprime os comandos de um ID de trabalho específico na saída padrão.

-d

Exclui trabalhos com base em seu ID de trabalho. É um alias para `atrm`.

-f

Lê o job em um arquivo em vez da entrada padrão.

-l

Lista as tarefas pendentes do usuário. Se o usuário for root, todos os trabalhos de todos os usuários serão listados. É um alias para `atq`.

-m

Envia um email para o usuário no final do trabalho, mesmo se não houver saída.

-q

Especifica uma fila na forma de uma única letra de a a z e de A a Z (por padrão, a para `at` e b para `batch`). Os jobs nas filas com as letras mais altas são executados com um valor nice maior. Os jobs enviados a uma fila com uma letra maiúscula são tratados como trabalhos em lote (`batch`).

-v

Mostra a hora em que o trabalho será executado antes de ler o trabalho.

Listar jobs programados com `atq`

Agora vamos agendar mais dois jobs `at`: o primeiro executa o script `foo.sh` às 09:30 e o segundo executa o script `bar.sh` após uma hora.

```
$ 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
```

Para listar os jobs pendentes, use o comando `atq`, que mostra as seguintes informações para cada job: ID do trabalho, data de execução do trabalho, tempo de execução do trabalho, fila e nome de usuário.

```
$ 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
```

Lembre-se de que o comando `at -l` é um alias para `atq`.

NOTE

Se você executar `atq` como root, ele exibirá os trabalhos na fila para todos os usuários.

Excluir jobs com `atrm`

Se quiser excluir um job `at`, use o comando `atrm` seguido do ID do trabalho. Por exemplo, para excluir o job com ID 14, rodamos o seguinte:

```
$ atrm 14
```

É possível excluir vários trabalhos com `atrm` especificando diversos IDs separados por espaços. Lembre-se de que o comando `at -d` é um alias de `atrm`.

NOTE

O usuário que rodar `atrm` como root pode excluir os jobs de todos os outros usuários.

Configurando o acesso ao agendamento de trabalhos

A autorização para usuários comuns agendarem jobs at é determinada pelos arquivos /etc/at.allow e /etc/at.deny. Se /etc/at.allow existir, somente os usuários não-root listados dentro dele podem agendar jobs at. Se /etc/at.allow não existir, mas /etc/at.deny existir, somente os usuários não-root listados dentro dele não podem agendar job at (neste caso, um arquivo /etc/at.deny vazio indica que todos os usuários podem agendar jobs at). Se nenhum desses arquivos existir, o acesso dos usuários ao agendamento de jobs at depende da distribuição usada.

Especificações de tempo

Para especificar quando um job at determinado deve ser executado, use o formato HH:MM, seguido opcionalmente por AM ou PM no caso do formato de 12 horas. Se o momento especificado já tiver passado, a instrução é aplicada no dia seguinte. Para agendar uma data em particular para executar o job, é preciso adicionar as informações de data após a hora usando um dos formatos a seguir: nome-do-mês dia-do-mês, nome-do-mês dia-do-mês ano, MMDDYY, MM/DD/YY, DD.MM.YY e YYYY-MM-DD).

As palavras-chave a seguir também são aceitas: midnight (meia-noite), noon (meio-dia), teatime (16h) e now (agora), seguidas por um sinal de mais (+) e um período de tempo (minutos, horas, dias e semanas). Finalmente, você pode dizer ao at para executar o trabalho hoje ou amanhã adicionando as palavras today ou tomorrow à hora determinada. Assim, usariamos at 07:15 AM Jan 01 para executar um trabalho às 07:15 da manhã de 1º de janeiro, e at now +5 minutes para executar um job daqui a cinco minutos. Leia o arquivo timespec na árvore /usr/share para saber mais sobre a definição exata das especificações de data e hora.

Uma alternativa ao at

Usando o systemd como gerenciador de sistema e serviços, também é possível agendar tarefas únicas com o comando `systemd-run`. Normalmente ele é usado para criar uma unidade transiente de temporizador para que um comando seja executado em um momento específico sem a necessidade de se criar um arquivo de serviço. Por exemplo, atuando como root, você pode executar o comando date às 11h30 em 06/10/2019 usando o seguinte:

```
# systemd-run --on-calendar='2019-10-06 11:30' date
```

Se quiser executar o script `foo.sh`, localizado em seu diretório atual, depois de dois minutos, use:

```
# systemctl-run --on-active="2m" ./foo.sh
```

Consulte as páginas de manual para aprender todos os usos possíveis de `systemd-run` com `systemd-run(1)`.

NOTE

Os temporizadores são registrados no diário do systemd e você pode rever os registros das diferentes unidades usando o comando `journalctl`. Além disso, se estiver trabalhando como um usuário comum, precisará usar a opção `--user` dos comandos `systemctl` e `journalctl`.

Exercícios Guiados

1. Para cada uma das seguintes especificações de tempo, indique qual é válida e qual é inválida para `at`:

`at 08:30 AM next week`

`at midday`

`at 01-01-2020 07:30 PM`

`at 21:50 01.01.20`

`at now +4 days`

`at 10:15 PM 31/03/2021`

`at tomorrow 08:30 AM`

2. Depois de agendar um trabalho com `at`, como você pode rever seus comandos?

3. Quais comandos você pode usar para rever seus jobs `at` pendentes? Quais comandos usaria para excluí-los?

4. Com o `systemd`, qual comando é usado como alternativa a `at`?

Exercícios Exploratórios

1. Crie um job `at` que execute o script `foo.sh`, localizado em seu diretório inicial, às 10h30 do próximo dia 31 de outubro. Suponha que você esteja agindo como um usuário comum.

2. Faça login no sistema como outro usuário comum e crie outro trabalho `at` que execute o script `bar.sh` amanhã às 10:00. Suponha que o script está localizado no diretório inicial do usuário.

3. Faça login no sistema como outro usuário comum e crie outro trabalho `at` que execute o script `foobar.sh` após 30 minutos. Suponha que o script está localizado no diretório inicial do usuário.

4. Logado como root, execute o comando `atq` para listar os jobs `at` programados de todos os usuários. O que acontece se um usuário comum executar esse comando?

5. Como root, exclua todos esses jobs `at` pendentes usando um único comando.

6. Como root, rode o comando `ls -l /usr/bin/at` e examine suas permissões.

Resumo

Nesta lição, você aprendeu:

- Usar `at` para rodar jobs únicos em um momento específico.
- Gerenciar jobs `at`.
- Configurar o acesso dos usuários ao agendamento de jobs `at`.
- Usar o `systemd-run` como alternativa ao `at`.

Os seguintes comandos e arquivos foram discutidos nesta lição:

`at`

Executa comandos em um momento especificado.

`atq`

Lista os trabalhos `at` pendentes do usuário, a menos que se trate do superusuário.

`atrm`

Exclui trabalhos `at`, identificados por seu número de trabalho.

`/etc/at.allow` and `/etc/at.deny`

Arquivos particulares usados para definir restrições `at`.

`systemd-run`

Cria e inicia uma unidade `timer` transiente como alternativa ao `at` para agendamentos únicos.

Respostas aos Exercícios Guiados

1. Para cada uma das seguintes especificações de tempo, indique qual é válida e qual é inválida para `at`:

<code>at 08:30 AM next week</code>	Válida
<code>at midday</code>	Inválida
<code>at 01-01-2020 07:30 PM</code>	Inválida
<code>at 21:50 01.01.20</code>	Válida
<code>at now +4 days</code>	Válida
<code>at 10:15 PM 31/03/2021</code>	Inválida
<code>at tomorrow 08:30 AM monotonic</code>	Inválida

2. Depois de agendar um trabalho com `at`, como você pode rever seus comandos?

Use o comando `at -c` seguido pelo ID do trabalho cujos comandos deseja rever. Observe que a saída também contém a maior parte do ambiente que estava ativo no momento em que o job foi agendado. Lembre-se de que o root pode rever as tarefas de todos os usuários.

3. Quais comandos você pode usar para rever seus jobs `at` pendentes? Quais comandos usaria para excluí-los?

Use o comando `at -l` para rever os jobs pendentes e o comando `at -d` para excluir seus jobs. `at -l` é um alias para `atq` e `at -d` é um alias para `atrm`. O root pode listar e excluir jobs de todos os usuários.

4. Com o systemd, qual comando é usado como alternativa a `at`?

O comando `systemd-run` pode ser usado como uma alternativa a `at` para agendar trabalhos únicos. Por exemplo, você pode usá-lo para executar comandos em um horário específico, definindo um *temporizador de calendário* ou um *temporizador monotônico* relativo a diferentes pontos de partida.

Respostas aos Exercícios Exploratórios

1. Crie um job `at` que execute o script `foo.sh`, localizado em seu diretório inicial, às 10h30 do próximo dia 31 de outubro. Suponha que você esteja agindo como um usuário comum.

```
$ 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
```

2. Faça login no sistema como outro usuário comum e crie outro trabalho `at` que execute o script `bar.sh` amanhã às 10:00. Suponha que o script está localizado no diretório inicial do usuário.

```
$ 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
```

3. Faça login no sistema como outro usuário comum e crie outro trabalho `at` que execute o script `foobar.sh` após 30 minutos. Suponha que o script está localizado no diretório inicial do usuário.

```
$ 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
```

4. Logado como root, execute o comando `atq` para listar os jobs `at` programados de todos os usuários. O que acontece se um usuário comum executar esse comando?

```
# 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
```

Se você executar o comando `atq` como root, todos os jobs `at` pendentes de todos os usuários

são listados. Se executá-lo como um usuário comum, apenas seus próprios jobs at pendentes são listados.

5. Como root, exclua todos esses jobs at pendentes usando um único comando.

```
# atrm 50 51 52
```

6. Como root, rode o comando ls -l /usr/bin/at e examine suas permissões.

```
# ls -l /usr/bin/at
-rwsr-sr-x 1 daemon daemon 43762 Dec 1 2015 /usr/bin/at
```

Nesta distribuição, o comando at tem definidos os bits SUID (o caractere s em vez do sinalizador executável para o proprietário) e SGID (o caractere s em vez do sinalizador executável para o grupo), o que indica que é executado com os privilégios do proprietário e do grupo do arquivo (daemon para ambos). É por isso que usuários comuns podem agendar tarefas com at.



107.3 Localização e internacionalização

Referência ao LPI objectivo

LPIC-1 version 5.0, Exam 102, Objective 107.3

Peso

3

Áreas chave de conhecimento

- Configurar idioma e variáveis de ambiente.
- Configurar fuso horário e variáveis de ambiente.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- /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



**Linux
Professional
Institute**

107.3 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	107 Tarefas Administrativas
Objetivo:	107.3 Localização e internacionalização
Lição:	1 de 1

Introdução

Todas as principais distribuições Linux podem ser configuradas para usar configurações de localização personalizadas. Essas configurações incluem definições relacionadas à região e ao idioma, como fuso horário, idioma da interface e codificação de caracteres, e podem ser modificadas durante a instalação do sistema operacional ou a qualquer momento depois disso.

Os aplicativos contam com variáveis de ambiente, arquivos de configuração do sistema e comandos para decidir o formato de data e hora e o idioma adequados a empregar; assim, a maioria das distribuições Linux compartilha uma maneira padronizada de ajustar as configurações de data, hora e localização. Esses ajustes são importantes não somente para aprimorar a experiência do usuário, mas também para garantir que o tempo dos eventos do sistema — importante, por exemplo, para relatar problemas relacionados à segurança — seja calculado corretamente.

Para serem capazes de representar qualquer texto escrito, independentemente da língua falada, os sistemas operacionais modernos precisam de um *padrão de codificação de caracteres* de referência, e os sistemas Linux não fogem à regra. Como os computadores só conseguem lidar com números, um caractere de texto nada mais é do que um número associado a um símbolo gráfico.

Plataformas de computação distintas podem associar valores numéricos distintos ao mesmo caractere e, portanto, um padrão comum de codificação de caracteres é necessário para torná-los compatíveis. Um documento de texto criado em um sistema só será legível em outro sistema se ambos concordarem quanto ao formato de codificação e aos números associados a cada caractere, ou pelo menos se souberem converter entre os dois padrões.

A natureza heterogênea das configurações de localização nos sistemas baseados em Linux resulta em diferenças sutis entre as distribuições. Apesar dessas diferenças, todas as distribuições compartilham as mesmas ferramentas e conceitos básicos para configurar os aspectos de internacionalização de um sistema.

Fusos horários

Os fusos horários são faixas distintas da superfície da Terra, aproximadamente proporcionais, abrangendo o equivalente a uma hora, ou seja, regiões do mundo que experimentam a mesma hora do dia no mesmo momento. Uma vez que não existe uma única longitude que possa ser considerada como o início do dia para o mundo inteiro, os fusos horários são relativos ao *meridiano primário*, onde o ângulo de longitude da Terra foi definido como 0. O tempo no primeiro meridiano é chamado de *Tempo Universal Coordenado*, por convenção abreviado para UTC. Por razões práticas, os fusos horários não seguem a distância longitudinal exata a partir do ponto de referência (o meridiano principal). Em vez disso, eles são artificialmente adaptados de forma a seguir as fronteiras dos países ou outras subdivisões significativas.

As subdivisões políticas são tão relevantes que os fusos horários recebem nomes que homenageiam algum agente geográfico importante daquela área específica, geralmente com base no nome de um grande país ou cidade dentro da zona. No entanto, os fusos horários são divididos de acordo com seu deslocamento de tempo em relação a UTC; esse deslocamento também pode ser usado para indicar o fuso em questão. O fuso horário *GMT-5*, por exemplo, indica uma região na qual o horário UTC está cinco horas à frente, ou seja, essa região está 5 horas atrás do UTC. Da mesma forma, o fuso horário *GMT+3* indica uma região na qual o horário UTC está três horas atrás. O termo *GMT*—de *Greenwich Mean Time*—é usado como sinônimo de UTC nos nomes de fusos horários que indicam a diferença de hora.

Uma máquina conectada pode ser acessada de diferentes partes do mundo, de forma que é aconselhável definir o relógio do hardware para UTC (o fuso horário *GMT+0*) e deixar a escolha do fuso horário para cada caso particular. Os serviços em nuvem, por exemplo, costumam ser configurados para usar UTC, o que pode ajudar a mitigar eventuais inconsistências entre o horário local e o horário dos clientes ou de outros servidores. Por outro lado, os usuários que abrem uma sessão remota no servidor podem querer usar seu fuso horário local. Assim, caberá ao sistema operacional configurar o fuso horário correto de acordo com cada caso.

Além da data e hora atuais, o comando `date` também exibe o fuso horário atualmente configurado:

```
$ date
Mon Oct 21 10:45:21 -03 2019
```

O deslocamento em relação a UTC é dado pelo valor `-03`, que indica que a hora exibida é três horas a menos que UTC. Portanto, o horário UTC está três horas à frente, e assim *GMT-3* é o fuso horário correspondente para as configurações de tempo especificadas. O comando `timedatectl`, disponível em distribuições que usam o `systemd`, mostra mais detalhes sobre a data e hora do sistema:

```
$ 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
```

Como vemos na entrada `Time zone`, nomes de fusos horários baseados em localidades — como `America/Sao_Paulo` — também são aceitos. O fuso horário padrão para o sistema é mantido no arquivo `/etc/timezone`, seja com o nome descritivo ou com a diferença de hora em relação a UTC. Os nomes genéricos de fusos horários indicados pela diferença em relação a UTC devem incluir `Etc` na primeira parte do nome. Assim, para definir o fuso horário padrão como *GMT+3*, o nome do fuso horário deve ser `Etc/GMT+3`:

```
$ cat /etc/timezone
Etc/GMT+3
```

Embora os nomes de fusos horários com base em localidades não exijam a indicação da diferença de hora para funcionar, eles não são tão fáceis de escolher. A mesma zona pode ter mais de um nome, sendo assim mais difícil de lembrar. Para mitigar este problema, o comando `tzselect` oferece um método interativo que guia o usuário na definição correta do fuso horário. O comando `tzselect` deve estar disponível por padrão em todas as distribuições Linux, já que é fornecido pelo pacote que contém utilitários necessários relacionados à Biblioteca GNU C.

O comando `tzselect` será útil, por exemplo, para um usuário que deseja identificar o fuso

horário da “Cidade de São Paulo” no “Brasil”. O `tzselect` começa perguntando qual a macrorregião do local desejado:

```
$ 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
```

A opção 2 refere-se a locais da América do Norte e do Sul, não necessariamente no mesmo fuso horário. Também é possível especificar o fuso horário com coordenadas geográficas ou com a notação de deslocamento, também conhecida como *formato Posix TZ*. A etapa seguinte é escolher o país:

```
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

O território brasileiro abrange quatro fusos horários; portanto, o nome do país por si só não basta para definir o fuso horário. Na etapa seguinte, o `tzselect` pede que o usuário especifique a região local:

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

Nem todos os nomes de localidade estão presentes, mas escolher a região mais próxima já basta. As informações fornecidas serão então usadas pelo `tzselect` para exibir o fuso horário correspondente:

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
```

O nome do fuso horário resultante, `America/Sao_Paulo`, também pode ser usado como o conteúdo de `/etc/timezone` para informar o fuso horário padrão para o sistema:

```
$ cat /etc/timezone
America/Sao_Paulo
```

Conforme declarado na saída de `tzselect`, a variável de ambiente `TZ` define o fuso horário da sessão do shell, qualquer que seja o fuso horário padrão do sistema. Se adicionarmos a linha `TZ='America/Sao_Paulo'; export TZ` ao arquivo `~/.profile`, `America / Sao_Paulo` se tornará o fuso horário das sessões futuras do usuário. A variável `TZ` também pode ser modificada temporariamente durante a sessão atual, a fim de exibir a hora em um fuso horário diferente:

```
$ env TZ='Africa/Cairo' date
Mon Oct 21 15:45:21 EET 2019
```

No exemplo, o comando `env` executa o comando fornecido em uma nova sessão de sub-shell com as mesmas variáveis de ambiente da sessão atual, exceto pela variável `TZ`, modificada pelo argumento `TZ='Africa/Cairo'`.

Horário de verão

Muitas regiões adotam o horário de verão—quando os relógios são normalmente adiantados em uma hora—em parte do ano, o que pode levar um sistema mal configurado a relatar a hora errada durante aquela estação do ano.

O arquivo `/etc/localtime` contém os dados usados pelo sistema operacional para ajustar seu relógio corretamente. Os sistemas Linux padrão têm arquivos para todos os fusos horários no diretório `/usr/share/zoneinfo/`, de modo que `/etc/localtime` é apenas um link simbólico para o arquivo de dados real dentro desse diretório. Os arquivos em `/usr/share/zoneinfo/` são organizados pelo nome do fuso horário correspondente, então o arquivo de dados para o fuso horário `America/Sao_Paulo` será `/usr/share/zoneinfo/America/Sao_Paulo`.

Como as definições de horário de verão podem mudar, é importante manter em dia os arquivos de `/usr/share/zoneinfo/`. O comando de atualização da ferramenta de gerenciamento de pacotes fornecida pela distribuição deve atualizá-los sempre que uma nova versão estiver disponível.

Idioma e codificação de caracteres

Os sistemas Linux podem funcionar com uma ampla variedade de idiomas e codificações de caracteres não ocidentais, definições conhecidas como *localidades* (em inglês, *locales*). A configuração de localidade mais básica é a definição da variável de ambiente `LANG`, a partir da qual a maioria dos programas do shell identificam o idioma a ser usado.

O conteúdo da variável `LANG` segue o formato `ab_CD`, onde `ab` é o código do idioma e `CD` é o código da região. O código do idioma deve seguir o padrão ISO-639 e o código da região, o padrão ISO-3166. Um sistema configurado para usar o português do Brasil, por exemplo, deve ter a variável `LANG` definida como `pt_BR.UTF-8`:

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

Conforme visto na saída deste exemplo, a variável `LANG` também contém a codificação de caracteres destinada ao sistema. O ASCII, abreviação de *American Standard Code for Information Interchange*, foi o primeiro padrão de codificação de caracteres amplamente usado para comunicação eletrônica. No entanto, como o ASCII tem uma gama muito limitada de valores numéricos disponíveis e foi baseado no alfabeto inglês, ele não contém caracteres usados por outros idiomas ou um conjunto extenso de símbolos não-alfabéticos. A codificação UTF-8 é um *Padrão Unicode* para os caracteres ocidentais comuns, além de diversos outros símbolos não convencionais. Conforme afirmado pelo *Unicode Consortium*, mantenedor do *Padrão Unicode*, ele deve ser adotado por padrão para garantir a compatibilidade entre as plataformas de computação:

O padrão Unicode atribui um número exclusivo a cada caractere, em qualquer plataforma, dispositivo, aplicativo ou idioma. Ele foi adotado por todos os provedores de software modernos e agora permite que os dados sejam transportados através de muitas plataformas, dispositivos e aplicativos diferentes sem corrupção. O suporte a Unicode forma a base para a representação de idiomas e símbolos em todos os principais sistemas operacionais, mecanismos de pesquisa, navegadores, laptops e smartphones — além da Internet e da World Wide Web (URLs, HTML, XML, CSS, JSON, etc.). (...) o Padrão Unicode e a disponibilidade de ferramentas de suporte estão entre as mais recentes tendências globais de tecnologia de software.

— The Unicode Consortium, What is Unicode?

Alguns sistemas ainda podem usar padrões definidos pelo ISO—como o padrão ISO-8859-1—para a codificação de caracteres não-ASCII. No entanto, esses padrões de codificação de caracteres devem ser preteridos em favor dos padrões de codificação Unicode. Todos os principais sistemas operacionais tendem a adotar o Unicode por padrão.

As configurações de localidade de todo o sistema são definidas no arquivo `/etc/locale.conf`. A variável `LANG` e outras variáveis relacionadas à localidade são atribuídas neste arquivo como uma variável de shell comum, por exemplo:

```
$ cat /etc/locale.conf
LANG=pt_BR.UTF-8
```

Os usuários podem empregar uma configuração de localidade personalizada redefinindo a variável de ambiente `LANG`. Isso pode ser feito apenas para a sessão atual ou para sessões futuras, adicionando a nova definição ao perfil Bash do usuário em `~/.bash_profile` ou `~/.profile`. Entretanto, até que o usuário efetue login, a localidade padrão do sistema ainda será usada por programas independentes do usuário, como a tela de login do gerenciador de exibição.

TIP O comando `localectl`, disponível em sistemas que empregam o `systemd` como gerenciador de sistema, também pode ser usado para consultar e alterar a localidade do sistema. Por exemplo: `localectl set-locale LANG=en_US.UTF-8`.

Além da variável `LANG`, outras variáveis de ambiente afetam aspectos específicos da localidade, como o símbolo monetário ou o separador de milhar correto para números:

LC_COLLATE

Define a ordem alfabética. Uma de suas finalidades é definir a ordem em que os arquivos e diretórios são listados.

LC_CTYPE

Define como o sistema tratará certos conjuntos de caracteres. Ele define, por exemplo, quais caracteres considerar como *maiúsculas* ou *minúsculas*.

LC_MESSAGES

Define o idioma para exibir as mensagens de programas (principalmente programas do GNU).

LC_MONETARY

Define a unidade monetária e o formato da moeda.

LC_NUMERIC

Define o formato numérico para valores não-monetários. Sua finalidade principal é definir os separadores de milhar e decimais.

LC_TIME

Define o formato de hora e data.

LC_PAPER

Define o tamanho padrão do papel.

LC_ALL

Sobrepõe todas as outras variáveis, incluindo LANG.

O comando `locale` mostra todas as variáveis definidas na configuração de localidade atual:

```
$ 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=
```

A única variável indefinida é LC_ALL, que pode ser usada para substituir temporariamente todas as outras configurações locais. O exemplo a seguir mostra como o comando `date`—sendo executado em um sistema configurado para a localidade pt_BR.UTF-8—modifica sua saída de forma a cumprir a nova variável LC_ALL:

```
$ 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
```

A modificação da variável `LC_ALL` fez com que ambas as abreviações de dia da semana e nome do mês fossem mostradas em inglês americano (`en_US`). Não é obrigatório, entretanto, definir a mesma localidade para todas as variáveis. É possível, por exemplo, ter a linguagem definida como `pt_BR` e o formato numérico (`LC_NUMERIC`) no padrão americano.

Algumas configurações de localização alteram a forma como os programas lidam com a ordem alfabética e formatos de numeração. Embora os programas convencionais geralmente sejam capazes de escolher corretamente uma localidade comum para essas situações, os scripts podem se comportar de forma inesperada ao tentar ordenar corretamente uma lista de itens em ordem alfabética, por exemplo. Por este motivo, recomenda-se definir a variável de ambiente `LANG` para a localidade comum `C`, como em `LANG=C`, para que o script produza resultados inequívocos, independentemente das definições de localização usadas no sistema onde é executado. A localidade `C` realiza apenas uma comparação simples de bytes e, portanto, também terá um desempenho melhor do que as outras.

Conversão de codificação

Um texto pode aparecer com caracteres ilegíveis quando exibido em um sistema com uma configuração de codificação de caracteres diferente do sistema onde o texto foi criado. O comando `iconv` ajuda a resolver este problema, convertendo o arquivo de sua codificação de caracteres original para a desejada. Por exemplo, para converter um arquivo de nome `original.txt` da codificação ISO-8859-1 no arquivo `converted.txt` utilizando a codificação UTF-8, o seguinte comando pode ser usado:

```
$ iconv -f ISO-8859-1 -t UTF-8 original.txt > converted.txt
```

A opção `-f ISO-8859-1` (ou `--from-code=ISO-8859-1`) define a codificação do arquivo original e a opção `-t UTF-8` (ou `--to-code=UTF-8`) define a codificação do arquivo convertido. Para listar todas as codificações suportadas pelo comando `iconv`, usamos o comando `iconv -l` ou `iconv --list`. Ao invés de usar o redirecionamento de saída, como no exemplo, a opção `-o converted.txt` ou `--output converted.txt` também faz o serviço.

Exercícios Guiados

1. Com base na seguinte saída do comando `date`, qual é o fuso horário do sistema na notação GMT?

```
$ date  
Mon Oct 21 18:45:21 +05 2019
```

2. Para qual arquivo o link simbólico `/etc/localtime` deve apontar a fim de tornar `Europe/Brussels` o fuso horário local padrão do sistema?

3. Os caracteres dos arquivos de texto podem não ser exibidos corretamente em um sistema com uma codificação de caracteres diferente da usada na criação do documento de texto. Como o `iconv` pode ser usado para converter o arquivo `old.txt`, codificado em `WINDOWS-1252`, no arquivo `new.txt`, que usa a codificação `UTF-8`?

Exercícios Exploratórios

1. Qual comando tornará Pacific/Auckland o fuso horário padrão para a sessão do shell atual?

2. O comando `uptime` mostra, entre outras coisas, a *carga média* do sistema em números fracionários. Ele usa as configurações de localidade atuais para decidir se o separador de casas decimais deve ser um ponto ou uma vírgula. Se, por exemplo, o local atual estiver definido como `de_DE.UTF-8` (o local padrão da Alemanha), o `uptime` usará uma vírgula como separador. Sabendo que no idioma inglês americano o ponto é usado como separador, qual comando fará com que o `uptime` exiba as frações usando um ponto ao invés de uma vírgula no resto da sessão atual?

3. O comando `iconv` substitui todos os caracteres que não pertencem ao conjunto de caracteres de destino por um ponto de interrogação. Se `/TRANSLIT` for anexado à codificação de destino, os caracteres não representados no conjunto de caracteres de destino serão substituídos (transliterados) por um ou mais caracteres de aparência semelhante. Como esse método pode ser usado para converter um arquivo de texto UTF-8 chamado `readme.txt` em um arquivo ASCII simples chamado `ascii.txt`?

Resumo

Esta lição aborda a maneira de configurar um sistema Linux para trabalhar com idiomas e configurações de data e hora personalizadas. Também abordamos conceitos e configurações de codificação de caracteres, fundamentais para exibir corretamente o conteúdo de texto. A lição abrange os seguintes tópicos:

- Como os sistemas Linux selecionam o idioma para exibir as mensagens do shell.
- Compreender como os fusos horários afetam a hora local.
- Como identificar o fuso horário apropriado e modificar as configurações do sistema de acordo.
- O que são as codificações de caracteres e como converter entre elas.

Os comandos e procedimentos abordados foram:

- Variáveis de ambiente relacionadas a localidade e tempo, como `LC_ALL`, `LANG` e `TZ`.
- `/etc/timezone`
- `/etc/localtime`
- `/usr/share/zoneinfo/`
- `locale`
- `tzselect`
- `timedatectl`
- `date`
- `iconv`

Respostas aos Exercícios Guiados

1. Com base na seguinte saída do comando `date`, qual é o fuso horário do sistema na notação GMT?

```
$ date  
Mon Oct 21 18:45:21 +05 2019
```

Trata-se do fuso horário Etc/GMT+5.

2. Para qual arquivo o link simbólico `/etc/localtime` deve apontar a fim de tornar `Europe/Brussels` o fuso horário local padrão do sistema?

O link `/etc/localtime` deve apontar para `/usr/share/zoneinfo/Europe/Brussels`.

3. Os caracteres dos arquivos de texto podem não ser exibidos corretamente em um sistema com uma codificação de caracteres diferente da usada na criação do documento de texto. Como o `iconv` pode ser usado para converter o arquivo `old.txt`, codificado em WINDOWS-1252, no arquivo `new.txt`, que usa a codificação UTF-8?

O comando `iconv -f WINDOWS-1252 -t UTF-8 -o new.txt old.txt` realizará a conversão desejada.

Respostas aos Exercícios Exploratórios

1. Qual comando tornará Pacific/Auckland o fuso horário padrão para a sessão do shell atual?

```
export TZ=Pacific/Auckland
```

2. O comando `uptime` mostra, entre outras coisas, a *carga média* do sistema em números fracionários. Ele usa as configurações de localidade atuais para decidir se o separador de casas decimais deve ser um ponto ou uma vírgula. Se, por exemplo, o local atual estiver definido como `de_DE.UTF-8` (o local padrão da Alemanha), o `uptime` usará uma vírgula como separador. Sabendo que no idioma inglês americano o ponto é usado como separador, qual comando fará com que o `uptime` exiba as frações usando um ponto ao invés de uma vírgula no resto da sessão atual?

O comando `export LC_NUMERIC=en_US.UTF-8` ou `export LC_ALL=en_US.UTF-8`.

3. O comando `iconv` substitui todos os caracteres que não pertencem ao conjunto de caracteres de destino por um ponto de interrogação. Se `//TRANSLIT` for anexado à codificação de destino, os caracteres não representados no conjunto de caracteres de destino serão substituídos (transliterados) por um ou mais caracteres de aparência semelhante. Como esse método pode ser usado para converter um arquivo de texto UTF-8 chamado `readme.txt` em um arquivo ASCII simples chamado `ascii.txt`?

O comando `iconv -f UTF-8 -t ASCII//TRANSLIT -o ascii.txt readme.txt` realizará a conversão desejada.



Tópico 108: Serviços essenciais do sistema



Linux
Professional
Institute

108.1 Manutenção da data e hora do sistema

Referência ao LPI objectivo

[LPIC-1 version 5.0, Exam 102, Objective 108.1](#)

Peso

3

Áreas chave de conhecimento

- Definir a data e a hora do sistema.
- Definir o relógio do hardware com a hora correta em UTC.
- Configurar o fuso horário correto.
- Configuração básica do NTP usando o ntpd e o chrony.
- Conhecimento de como usar o serviço pool.ntp.org.
- Noções do comando ntpq.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- /usr/share/zoneinfo/
- /etc/timezone
- /etc/localtime
- /etc/ntp.conf
- /etc/chrony.conf
- date
- hwclock
- timedatectl

- ntpd
- ntpdate
- chronyc
- pool.ntp.org



**Linux
Professional
Institute**

108.1 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	108 Serviços essenciais do sistema
Objetivo:	108.1 Controlar a hora do sistema
Lição:	1 de 2

Introdução

Na computação moderna, é essencial manter um controle preciso do tempo. A implementação desse controle, no entanto, é surpreendentemente complexa. A prática de controlar a hora parece trivial para o usuário final, mas o sistema precisa ser capaz de lidar com muitas idiossincrasias e casos extremos de forma inteligente. É preciso levar em conta o fato de que os fusos horários não são estáticos, mas podem ser alterados por uma decisão administrativa ou política. Um país pode optar por acabar com o horário de verão. Qualquer programa deve ser capaz de lidar com essas mudanças de uma maneira lógica. Felizmente para os administradores de sistema, as soluções para o controle de data e hora no sistema operacional Linux são maduras, robustas e geralmente funcionam sem muita interferência.

Quando um computador Linux é inicializado, ele começa a marcar o tempo. Nós nos referimos a isso como o *relógio do sistema*, uma vez que é atualizado pelo sistema operacional. Além disso, os computadores modernos também têm um *relógio de hardware ou de tempo real*. Esse relógio de hardware é um recurso da placa-mãe que marca o tempo, esteja o computador funcionando ou não. Durante a inicialização, a hora do sistema é definida a partir do relógio de hardware, mas na maioria das vezes esses dois relógios funcionam independentemente um do outro. Nesta lição, discutiremos os métodos de interação com o relógio do sistema e o do hardware.

Na maioria dos sistemas Linux modernos, a hora do sistema e a hora do hardware são sincronizadas com a *hora da rede*, implementada pelo *Network Time Protocol* (NTP). Na grande maioria dos casos, a única configuração que um usuário normal precisa fazer é definir seu fuso horário; o NTP cuida do resto. No entanto, falaremos de algumas maneiras de ajustar a hora manualmente. Pontos específicos da configuração da hora da rede serão abordados na próxima lição.

Hora local e hora universal

O relógio do sistema está configurado para o Tempo Universal Coordenado (UTC), que é a hora local de Greenwich, Reino Unido. Normalmente, um usuário deseja saber sua *hora local*. A hora local é calculada tomando-se a hora UTC e aplicando um *deslocamento* com base no fuso horário e no horário de verão. Desta forma, evitam-se muitos cálculos complexos.

O relógio do sistema pode ser definido para a hora UTC ou local, mas é recomendável que também seja definido para a hora UTC.

Date

`date` é um utilitário nativo que simplesmente imprime a hora local:

```
$ date  
Sun Nov 17 12:55:06 EST 2019
```

Se modificarmos as opções do comando `date`, mudamos o formato da saída.

Por exemplo, `date -u` serve para exibir o horário UTC atual.

```
$ date -u  
Sun Nov 17 18:02:51 UTC 2019
```

Outras opções comumente usadas retornam a hora local em um formato RFC aceito:

-I

Data/hora no formato ISO 8601. Anexar `date (-I)` limita a saída apenas à data. Outros formatos são `hours`, `minutes`, `seconds` e `ns` (para nanossegundos).

-R

Retorna data e hora no formato RFC 5322.

--rfc-3339

Retorna data e hora no formato RFC 3339.

O formato de `date` pode ser personalizado pelo usuário com as sequências especificadas na página de manual. Por exemplo, a hora atual pode ser formatada como tempo do Unix da seguinte maneira:

```
$ date +%
1574014515
```

Na página de manual de `date`, podemos ver que `%s` se refere ao tempo do Unix.

O tempo do Unix é usado internamente na maioria dos sistemas baseados em Unix. Ele armazena a hora UTC como o número de segundos desde a *Época*, que foi definida como 1º de janeiro de 1970.

NOTE

O número de bits necessários para armazenar a hora Unix no momento atual é de 32 bits. No futuro, 32 bits se tornarão insuficientes para conter a hora atual no formato Unix. Isso causará problemas sérios para qualquer sistema Linux de 32 bits. Felizmente, isso não ocorrerá até 19 de janeiro de 2038.

Usando essas sequências, podemos formatar a data e a hora em quase todos os formatos exigidos por qualquer aplicativo. Claro, na maioria dos casos é preferível seguir um padrão aceito.

Além disso, `date --date` pode ser usado para formatar um horário que não seja a hora atual. Neste caso, um usuário pode especificar a data a ser aplicada ao sistema usando o tempo do Unix, por exemplo:

```
$ date --date='@1564013011'
Wed Jul 24 20:03:31 EDT 2019
```

A opção `--debug` é muito útil para garantir que uma data possa ser analisada com sucesso. Observe o que acontece ao passarmos uma data válida para o comando:

```
$ 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)
```

Essa é uma ferramenta útil para a resolução de problemas em um aplicativo que gera uma data.

Relógio do hardware

O comando `hwclock` exibe a hora mantida no relógio de tempo real. Este comando exige privilégios elevados; portanto, usamos `sudo` para chamar o comando neste caso:

```
$ sudo hwclock
2019-11-20 11:31:29.217627-05:00
```

A opção `--verbose` retorna mais resultados, que podem ser úteis para solucionar problemas:

```
$ 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
```

Observe o `Calculated Hardware Clock drift`. Esta saída informa se a hora do sistema e a hora do hardware estão divergindo uma da outra.

timedatectl

`timedatectl` é um comando que pode ser usado para verificar o status geral de hora e data, incluindo se a hora da rede foi ou não sincronizada (na próxima lição, trataremos do Network

Time Protocol).

Por padrão, `timedatectl` retorna informações semelhantes a `date`, mas com a adição da hora RTC (hardware), bem como o status do serviço 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
```

Configurando a hora com `timedatectl`

Se o NTP não estiver disponível, recomenda-se usar `timedatectl` em vez de `date` ou `hwclock` para definir a hora:

```
# timedatectl set-time '2011-11-25 14:00:00'
```

O processo é semelhante ao de `date`. O usuário também pode definir a hora independentemente da data com o formato HH:MM:SS.

Definindo o fuso horário com `timedatectl`

`timedatectl` é a melhor maneira de configurar o fuso horário local nos sistemas Linux baseados em `systemd` quando não existe GUI. `timedatectl` lista os fusos horários possíveis e, a partir daí, o fuso horário pode ser definido usando um deles como argumento.

Primeiro, listamos os fusos horários possíveis:

```
$ timedatectl list-timezones
Africa/Abidjan
Africa/Accra
Africa/Algiers
Africa/Bissau
Africa/Cairo
...
```

A lista de fusos horários possíveis é longa, por isso o uso do comando `grep` é recomendado neste caso.

Em seguida, podemos definir o fuso horário usando um dos elementos da lista retornada:

```
$ 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
```

Lembre-se de que o nome do fuso horário deve ser exato. `Africa/Cairo`, por exemplo, muda o fuso horário, mas `cairo` ou `africa/cairo` não.

Desativando o NTP com `timedatectl`

Em alguns casos, pode ser necessário desativar o NTP. Podemos fazer isso com `systemctl`, mas vamos demonstrar o procedimento com `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
```

Definindo o fuso horário sem `timedatectl`

A definição do fuso horário é uma etapa padrão ao se instalar o Linux em uma nova máquina. Se houver um processo de instalação gráfico, isso provavelmente será feito sem nenhuma ação adicional do usuário.

O diretório `/usr/share/zoneinfo` contém informações sobre os diferentes fusos horários

possíveis. No diretório `zoneinfo`, há subdiretórios com o nome dos continentes, bem como outros links simbólicos. Recomenda-se encontrar o `zoneinfo` da sua região começando por seu continente.

Os arquivos `zoneinfo` contêm as regras necessárias para calcular a diferença de horário local em relação a UTC e também são importantes se a sua região segue o horário de verão. O conteúdo de `/etc/localtime` será lido quando o Linux precisar determinar o fuso horário local. Para definir o fuso horário sem o uso de uma GUI, o usuário deve criar um link simbólico de `/usr/share/zoneinfo` para `/etc/localtime` informando sua localização. Por exemplo:

```
$ ln -s /usr/share/zoneinfo/Canada/Eastern /etc/localtime
```

Depois de definir o fuso horário correto, recomenda-se executar:

```
# hwclock --systohc
```

Isso configurará o *relógio do hardware* a partir do *relógio do sistema* (ou seja, o relógio em tempo real será configurado para a mesma hora que `date`). Note que este comando é executado com privilégios de root; neste caso, você está logado como root.

`/etc/timezone` é semelhante a `/etc/localtime`. É uma representação de dados do fuso horário local e, como tal, pode ser lido usando `cat`:

```
$ cat /etc/timezone
America/Toronto
```

Observe que este arquivo não é usado por todas as distribuições Linux.

Configurando data e hora sem `timedatectl`

NOTE

A maioria dos sistemas Linux modernos usa o `systemd` para sua configuração e serviços; assim, não é recomendado usar `date` ou `hwclock` para definir a hora. O `systemd` emprega para isso o `timedatectl`. No entanto, é importante conhecer esses comandos legados no caso de você precisar administrar um sistema mais antigo.

Usando `date`

`date` tem uma opção para definir a hora do sistema. Use `--set` ou `-s` para definir a data e hora.

Também há a opção `--debug` para verificar a análise correta do comando:

```
# date --set="11 Nov 2011 11:11:11"
```

Neste caso, é necessário ter privilégios de root para definir a data. Também podemos optar por alterar a hora ou data independentemente:

```
# date +%Y%m%d -s "20111125"
```

Aqui, devemos especificar as sequências para que nossa string seja analisada corretamente. Por exemplo, `%Y` refere-se ao ano e, portanto, os primeiros quatro dígitos `2011` serão interpretados como o ano de 2011. Da mesma forma, `%T` é a sequência de hora, como demonstrado aqui ao definirmos a hora:

```
# date +%T -s "13:11:00"
```

Depois de alterar a hora do sistema, é recomendável também definir o relógio do hardware para que os relógios do sistema e do hardware estejam sincronizados:

```
# hwclock --systohc
```

`systohc` significa “relógio do sistema para relógio do hardware”.

Usando `hwclock`

Em vez de definir o relógio do sistema e atualizar o relógio do hardware, podemos optar por reverter o processo. Começaremos ajustando o relógio do hardware:

```
# hwclock --set --date "4/12/2019 11:15:19"
# hwclock
Fri 12 Apr 2019 6:15:19 AM EST -0.562862 seconds
```

Note que, por padrão, `hwclock` espera a hora UTC, mas retorna a hora local.

Depois de configurar o relógio do hardware, precisamos atualizar o relógio do sistema a partir dele. `hctosys` pode ser interpretado como “relógio do hardware para relógio do sistema”.

```
# hwclock --hctosys
```

Exercícios Guiados

1. Indique se os comandos a seguir estão exibindo ou modificando a *hora do sistema* ou a *hora do hardware*:

Comando(s)	Sistema	Hardware	Ambos
date -u			
hwclock --set --date "12:00:00"			
timedatectl			
timedatectl grep RTC			
hwclock --hctosys			
date +%T -s "08:00:00"			
timedatectl set- time 1980-01-10			

2. Observe a seguinte saída e, em seguida, corrija o formato do argumento para que o comando seja bem-sucedido:

```
$ 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. Use o comando `date` e as sequências para que o mês do sistema seja definido como fevereiro. Deixe o resto da data e hora inalterados.

4. Pressupondo que o comando acima foi bem-sucedido, use `hwclock` para ajustar o relógio do hardware a partir do relógio do sistema.

5. Existe um local chamado `eucla`. De que continente ele faz parte? Use o comando `grep` para descobrir.

6. Defina seu fuso horário atual para `eucla`.

Exercícios Exploratórios

1. Qual método de configuração de hora é o ideal? Em que caso o método preferido seria impraticável?

2. Por que você acha que existem tantos métodos para realizar a mesma coisa, ou seja, definir a hora do sistema?

3. Após 19 de janeiro de 2038, o Tempo do Linux exigirá um número de 64 bits para ser armazenado. No entanto, poderíamos simplesmente ter optado por definir uma “Nova Época”. Por exemplo, 1º de janeiro de 2038 à meia-noite pode ser definido como a Hora 0 de uma Nova Época. Por que você acha que esta solução não foi adotada?

Resumo

Nesta lição, você aprendeu:

- Como exibir a hora em diferentes formatos a partir da linha de comando.
- A diferença entre o relógio do sistema e o relógio do hardware no Linux.
- Como ajustar manualmente o relógio do sistema.
- Como ajustar manualmente o relógio do hardware.
- Como alterar o fuso horário do sistema.

Comandos usados nesta lição:

date

Exibe ou altera o relógio do sistema. Outras opções:

-u

Exibe a hora UTC.

+%s

Usa uma sequência para exibir a hora da Época.

--date=

Determina uma hora específica a exibir, sem ser a hora atual.

--debug

Exibe mensagens de depuração ao analisar uma data inserida pelo usuário.

-s

Ajusta o relógio do sistema manualmente.

hwclock

Exibe ou altera o relógio do hardware.

--systohc

Usa o relógio do sistema para definir o relógio do hardware.

--hctosys

Usa o relógio do hardware para definir o relógio do sistema.

--set --date

Ajusta o relógio do hardware manualmente.

timedatectl

Exibe os relógios do sistema e do hardware, bem como a configuração NTP em sistemas Linux baseados em systemd.

set-time

Define a hora manualmente.

list-timezones

Lista os fusos horários possíveis.

set-timezone

Define o fuso horário manualmente.

set-ntp

Ativa/desativa o NTP.

Respostas aos Exercícios Guiados

1. Indique se os comandos a seguir estão exibindo ou modificando a *hora do sistema* ou a *hora do hardware*:

Comando(s)	Sistema	Hardware	Ambos
date -u	X		
hwclock --set --date "12:00:00"		X	
timedatectl			X
timedatectl grep RTC		X	
hwclock --hctosys	X		
date +%T -s "08:00:00"	X		
timedatectl set- time 1980-01-10			X

2. Observe a seguinte saída e, em seguida, corrija o formato do argumento para que o comando seja bem-sucedido:

```
$ 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. Use o comando `date` e as sequências para que o mês do sistema seja definido como fevereiro. Deixe o resto da data e hora inalterados.

```
date +%m -s "2"
```

4. Pressupondo que o comando acima foi bem-sucedido, use `hwclock` para ajustar o relógio do hardware a partir do relógio do sistema.

```
hwclock -systohc
```

5. Existe um local chamado `eucla`. De que continente ele faz parte? Use o comando `grep` para descobrir.

```
timedatectl list-timezones \| grep -i eucla
```

OU

```
grep -ri eucla /usr/share/zoneinfo
```

6. Defina seu fuso horário atual para `eucla`.

```
timedatectl set-timezone 'Australia/Eucla'
```

ou

```
ln -s /usr/share/zoneinfo/Australia/Eucla /etc/localtime
```

Respostas aos Exercícios Exploratórios

1. Qual método de configuração de hora é o ideal? Em que caso o método preferido seria impraticável?

Na maioria das distribuições Linux, o NTP vem habilitado por padrão e define a hora do sistema sem interferência do usuário. No entanto, se um sistema Linux não estiver conectado à internet, o NTP ficará inacessível. Por exemplo, um sistema Linux embarcado rodando em um equipamento industrial pode não ter conectividade de rede.

2. Por que você acha que existem tantos métodos para realizar a mesma coisa, ou seja, definir a hora do sistema?

Como a configuração da hora tem sido um requisito de todos os sistemas *nix por décadas, existem muitos métodos legados para fazer essa configuração e que ainda são mantidos.

3. Após 19 de janeiro de 2038, o Tempo do Linux exigirá um número de 64 bits para ser armazenado. No entanto, poderíamos simplesmente ter optado por definir uma “Nova Época”. Por exemplo, 1º de janeiro de 2038 à meia-noite pode ser definido como a Hora 0 de uma Nova Época. Por que você acha que esta solução não foi adotada?

Até 2038, a grande maioria dos computadores já estará equipado de CPUs de 64 bits, e usar um número de 64 bits não degradará o desempenho de forma significativa. No entanto, seria impossível estimar os riscos de “reiniciar” o tempo da Época dessa maneira. Muitos softwares legados poderiam ser afetados. Os bancos e grandes empresas, por exemplo, costumam ter uma grande quantidade de programas antigos dos quais dependem para uso interno. Portanto, esse cenário, como tantos outros, exigia que se encontrasse um meio-termo. Qualquer sistema de 32 bits ainda em execução em 2038 sofreria com o estouro do Tempo da Época, mas o software legado seria impactado pela alteração do valor da Época.



108.1 Lição 2

Certificação:	LPIC-1
Versão:	5.0
Tópico:	108 Serviços essenciais do sistema
Objetivo:	108.1 Manutenção da data e hora do sistema
Lição:	2 de 2

Introdução

Embora os computadores pessoais sejam capazes de manter a hora de maneira razoavelmente precisa por conta própria, os sistemas de produção e os ambientes de rede exigem um controle extremamente exato do tempo. A medição de tempo mais precisa é realizada por *relógios de referência*, normalmente relógios atômicos. No mundo moderno, todos os sistemas informáticos conectados à Internet podem ser sincronizados com esses relógios de referência usando o chamado *Network Time Protocol* (NTP). Um sistema informático com NTP será capaz de sincronizar os relógios do sistema com a hora fornecida pelos relógios de referência. Se a hora do sistema e a hora medida nesses servidores forem diferentes, o computador aumentará ou diminuirá a hora do sistema interno gradativamente até que a hora do sistema corresponda à hora da rede.

O NTP emprega uma estrutura hierárquica para divulgar o tempo. Os relógios de referência são conectados a servidores no topo da hierarquia. Esses servidores são máquinas *Estrato 1* e, normalmente, não são acessíveis ao público. As máquinas Estrato 1 são, entretanto, acessíveis a máquinas *Estrato 2*, por sua vez acessíveis a máquinas *Estrato 3* e assim por diante. Os servidores Estrato 2 são acessíveis ao público, assim como qualquer máquina inferior na hierarquia. Ao configurar o NTP para uma rede grande, é recomendável conectar um pequeno número de

computadores aos servidores Estrato 2+ e, a partir daí, fazer com que essas máquinas forneçam o NTP a todas as outras máquinas. Desta forma, minimizam-se as demandas sobre as máquinas Estrato 2.

Alguns termos são importantes ao se falar em NTP. Alguns desses termos aparecem nos comandos que usamos para rastrear o status do NTP em nossas máquinas:

Deslocamento (offset)

Refere-se à diferença absoluta entre a hora do sistema e a hora NTP. Por exemplo, se o relógio do sistema marca 12:00:02 e o horário NTP é 11:59:58, o deslocamento entre os dois relógios é de quatro segundos.

Salto (step)

Se o deslocamento de tempo entre o provedor NTP e um consumidor for maior que 128ms, o NTP executará uma única alteração significativa na hora do sistema, em vez de desacelerar ou acelerar o relógio do sistema. Isso é chamado de *stepping*.

Ajuste gradativo (slew)

Slew refere-se às alterações feitas na hora do sistema quando o deslocamento entre a hora do sistema e o NTP é menor que 128ms. Se esse for o caso, as alterações serão feitas gradualmente. Isso é conhecido como *slewing*.

Relógio insano

Se o deslocamento entre a hora do sistema e a hora NTP for maior que 17 minutos, o tempo do sistema é considerado *insano* e o daemon NTP não introduzirá nenhuma alteração no relógio do sistema. Será preciso tomar medidas especiais para trazer a hora do sistema até menos de 17 minutos da hora correta.

Escorregamento ou deslizamento (drift)

O escorregamento se refere ao fenômeno em que dois relógios ficam fora de sincronia com o tempo. Essencialmente, se dois relógios são inicialmente sincronizados, mas vão ficando fora de sincronia com o passar do tempo, está ocorrendo um escorregamento do relógio.

Variação (jitter)

A variação refere-se à quantidade de escorregamento desde a última vez em que um relógio foi consultado. Portanto, se a última sincronização com o NTP ocorreu há 17 minutos, e o deslocamento entre o provedor NTP e o consumidor é de 3 milissegundos, então 3 milissegundos é a variação.

Agora vamos discutir algumas das maneiras específicas como o Linux implementa o NTP.

timedatectl

Se sua distribuição Linux usa `timedatectl`, por padrão ela implementa um cliente *SNTP* ao invés de uma implementação NTP completa. Esta é uma implementação menos complexa de tempo de rede e significa que a máquina não servirá NTP para outros computadores conectados.

Neste caso, o *SNTP* não funcionará a menos que o serviço `timesyncd` esteja rodando. Como acontece com todos os serviços `systemd`, podemos verificar se ele está rodando com:

```
$ 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
      CGrou: /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).
...
...
```

O status da sincronização *SNTP* `timedatectl` pode ser verificado com `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
```

Essa configuração pode ser adequada para a maioria das situações; porém, conforme já observamos, ela é insuficiente quando se deseja sincronizar diversos clientes em uma rede. Nesse caso, é recomendável instalar um cliente NTP completo.

Daemon NTP

A hora do sistema é regularmente comparada à hora da rede. Para que isso funcione, é necessário ter um *daemon* rodando em segundo plano. Em muitos sistemas Linux, o nome desse daemon é `ntpd`. O `ntpd` permite que uma máquina não seja apenas um *consumidor de tempo* (isto é, capaz de sincronizar seu próprio relógio a partir de uma fonte externa), mas também *forneca tempo* para outras máquinas.

Vamos supor que nosso computador seja baseado no `systemd` e use `systemctl` para controlar os daemons. Vamos instalar os pacotes `ntp` usando o gerenciador de pacotes apropriado e, em seguida, verificar se o daemon `ntpd` está rodando:

```
$ systemctl status ntpd

● ntpd.service - Network Time Service
  Loaded: loaded (/usr/lib/systemd/system/ntp.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/ntp.service
             `--867 /usr/sbin/ntpd -u ntp:ntp -g
```

Em alguns casos, pode ser necessário iniciar e habilitar o `ntpd`. Na maioria das máquinas Linux, isso é feito com:

```
# systemctl enable ntpd && systemctl start ntpd
```

As consultas NTP acontecem na porta TCP 123. Se o NTP falhar, verifique essa porta está aberta e escutando.

Configuração do NTP

O NTP pode pesquisar várias fontes e selecionar os melhores candidatos a serem usados para definir a hora do sistema. Se uma conexão de rede for perdida, o NTP utiliza os ajustes anteriores de seu histórico para estimar os ajustes futuros.

Dependendo da distribuição do Linux, a lista de servidores de hora da rede será armazenada em locais diferentes. Vamos supor que `ntp` esteja instalado em sua máquina.

O arquivo `/etc/ntp.conf` contém informações de configuração sobre como o sistema se sincroniza ao tempo da rede. Esse arquivo pode ser lido e modificado com `vi` ou `nano`.

Por padrão, os servidores NTP usados serão especificados em uma seção como esta:

```
# 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
```

A sintaxe para adicionar servidores NTP é assim:

```
server (IP Address)  
server server.url.localhost
```

Os endereços de servidor podem ser endereços IP ou URLs se o DNS tiver sido configurado corretamente. Nesse caso, o servidor sempre será consultado.

Um administrador de rede também pode considerar o uso (ou configuração) de um *pool*. Neste caso, pressupomos que existam diversos provedores NTP, todos executando daemons NTP e com o mesmo tempo. Quando um cliente consulta um pool, um provedor é selecionado aleatoriamente. Isso ajuda a distribuir a carga da rede entre muitas máquinas, de forma que nenhuma máquina do pool fique encarregada de todas as consultas de NTP.

Em geral `/etc/ntp.conf` estará equipado de um pool de servidores chamado `pool.ntp.org`. Assim, por exemplo, `server 0.centos.pool.ntp.org` é um pool NTP padrão fornecido para as máquinas CentOS.

pool.ntp.org

Os servidores NTP usados por padrão são um projeto de código aberto. Mais informações podem ser encontradas em ntppool.org.

Considere se o uso de do NTP Pool é apropriado para o seu caso. Se a empresa, a organização ou uma vida humana dependem de um horário correto ou podem ser prejudicados se ele estiver incorreto, você não deve apenas “baixar um negócio da internet”. O NTP Pool é geralmente de altíssima qualidade, mas é um serviço executado por voluntários em seu tempo livre. Fale com seus fornecedores de equipamentos e de serviço sobre como obter um serviço local e confiável configurado para você. Consulte também nossos termos de serviço. Recomendamos os servidores de horário da Meinberg, mas você também pode encontrar servidores de horário da End Run, Spectracom e muitos outros.

— ntppool.org

ntpdate

Durante a configuração inicial, a hora do sistema e o NTP podem ser seriamente dessincronizados. Se o *deslocamento* entre o sistema e a hora NTP for maior que 17 minutos, o daemon NTP não fará alterações na hora do sistema. Neste caso, será necessária uma intervenção manual.

Primeiramente, se `ntpd` estiver rodando, será necessário *interromper* o serviço. Use `systemctl stop ntpd` para fazer isso.

Em seguida, use `ntpdate pool.ntp.org` para realizar uma sincronização inicial única, onde `pool.ntp.org` se refere ao endereço IP ou URL de um servidor NTP. Pode ser necessária mais de uma sincronização.

ntpq

`ntpq` é um utilitário para monitorar o status do NTP. Uma vez que o daemon NTP foi iniciado e configurado, usamos `ntpq` para verificar seu status:

```
$ 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
+ntp2.0x00.1v    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
```

Neste caso, `-p` significa *imprimir*, ou seja, exibir um resumo dos pares. Os endereços de host também podem ser retornados como endereços IP usando `-n`.

remote

nome do host do provedor NTP.

refid

ID de referência do provedor NTP.

st

Estrato do provedor.

when

Número de segundos desde a última consulta.

poll

Número de segundos entre as consultas.

reach

ID de status para indicar se um servidor foi alcançado. As conexões bem-sucedidas aumentam este número em 1.

delay

Tempo em ms entre a consulta e a resposta do servidor.

offset

Tempo em ms entre a hora do sistema e a hora NTP.

jitter

Deslocamento em ms entre a hora do sistema e o NTP na última consulta.

`ntpq` também inclui um modo interativo, que é acessado quando ele é executado sem opções ou argumentos. A opção `?` retorna uma lista de comandos reconhecidos pelo `ntpq`.

chrony

`chrony` é outra forma de implementar o NTP. Ele é instalado por padrão em alguns sistemas Linux, mas está disponível para download em todas as principais distribuições. `chronyd` é o daemon `chrony` e `chronyc` é a interface de linha de comando. Pode ser necessário iniciar e habilitar `chronyd` antes de interagir com `chronyc`.

Se a instalação do chrony tiver uma configuração padrão, o uso do comando `chronyc tracking` fornecerá informações sobre o NTP e a hora do sistema:

\$ `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
```

Essa saída contém muitas informações, mais do que as que estão disponíveis em outras implementações.

Reference ID

O ID de referência e o nome ao qual o computador está sincronizado no momento.

Stratum

Número de passos até um computador com um relógio de referência anexado.

Ref time

Esta é a hora UTC em que a última medição da fonte de referência foi feita.

System time

Atraso do relógio do sistema do servidor sincronizado.

Last offset

Deslocamento estimado da última atualização do relógio.

RMS offset

Média de longo prazo do valor do deslocamento.

Frequency

Esta é a taxa na qual o relógio do sistema estaria errado se o chronyd não o estivesse

corrigindo. É fornecida em ppm (partes por milhão).

Residual freq

Frequência residual indicando a diferença entre as medições da fonte de referência e a frequência atualmente sendo usada.

Skew

Límite de erro estimado da frequência.

Root delay

Total de atrasos do caminho de rede até o computador do estrato a partir do qual o computador está sendo sincronizado.

Leap status

Este é o status de intercalação, que pode ter um dos seguintes valores: normal, inserir segundo, excluir segundo ou não sincronizado.

Também podemos ver informações detalhadas sobre a última atualização válida do NTP:

```
# 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
```

Finalmente, `chronyc sources` retorna informações sobre os servidores NTP usados para sincronizar a hora:

```
$ chronyc sources
210 Number of sources = 0
MS Name/IP address      Stratum Poll Reach LastRx Last sample
=====
```

No momento, esta máquina não possui fontes configuradas. Podemos adicionar fontes de `pool.ntp.org` abrindo o arquivo de configuração do `chrony`. Geralmente, ele fica em `/etc/chrony.conf`. Quando abrimos este arquivo, podemos ver que alguns servidores estão listados por padrão:

```
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
```

Esses servidores também servirão como um guia de sintaxe ao inserir nossos próprios servidores. No entanto, neste caso, simplesmente removeremos os `!` no início de cada linha, fazendo com que deixem de ser comentários, para usar os servidores padrão do projeto `pool.ntp.org`.

Além disso, neste arquivo, podemos optar por alterar a configuração padrão em relação ao skew (diferença entre a frequência real e a ideal) e ao escorregamento, bem como a localização do driftfile e do keyfile.

Nesta máquina, precisamos fazer uma grande correção inicial do relógio. Optamos por remover o comentário da seguinte linha:

```
! makestep 1.0 3
```

Depois de fazer alterações no arquivo de configuração, reinicie o serviço `chronyd` e use `chronyc makestep` para ajustar manualmente o relógio do sistema:

```
# chronyc makestep  
200 OK
```

E então use `chronyc tracking` como anteriormente para verificar se as alterações foram realizadas.

Exercícios Guiados

1. Insira o termo apropriado para cada definição:

Definição	Termo
Um computador que compartilha o tempo de rede com você	
A distância de um relógio de referência, em passos ou saltos	
A diferença entre o tempo do sistema e o tempo da rede	
A diferença entre o tempo do sistema e o tempo da rede desde a última consulta ao NTP	
Grupo de servidores que fornecem tempo de rede e compartilham a carga entre si	

2. Especifique os comandos usados para gerar os seguintes valores:

Valor	chronyc tracking	timedatectl show-timesync --all	ntpq -pn	chrony ntpdata	chronyc sources
Variação					
Escorregamento					
Intervalo de Consulta					
Deslocamento					
Estrato					
Endereço IP do provedor					
Atraso da raiz					

3. Você está configurando uma rede corporativa que consiste em um servidor Linux e diversos desktops Linux. O servidor tem o endereço IP estático 192.168.0.101. Você decide que o servidor

se conectará a pool.ntp.org e, em seguida, fornecerá o tempo NTP aos computadores pessoais. Descreva a configuração do servidor e dos computadores.

4. Uma máquina Linux está com a hora incorreta. Descreva as etapas que você executaria para solucionar os problemas de NTP.

Exercícios Exploratórios

1. Pesquise as diferenças entre SNTP e NTP.

SNTP	NTP

2. Por que um administrador de sistema optaria por *não* usar pool.ntp.org?

3. Como um administrador de sistema escolheria participar ou contribuir de outra forma com o projeto pool.ntp.org?

Resumo

Nesta lição, você aprendeu:

- O que é NTP e por que é importante.
- Configurar o daemon NTP do projeto pool.ntp.org.
- Usar ntpq para verificar a configuração de NTP.
- Usar chrony como serviço de NTP alternativo.

Comandos usados nesta lição:

timedatectl show-timesync --all

Exibe informações de SNTP se timedatectl estiver sendo usado.

ntpdate <address>

Realiza uma atualização manual única do NTP.

ntpq -p

Imprime um histórico das consultas recentes de NTP. -n substitui as URLs por endereços IP.

chronyc tracking

Exibe o status do NTP se o chrony estiver sendo usado.

chronyc ntpdata

Exibe informações de NTP sobre a última consulta.

chronyc sources

Exibe informações sobre provedores NTP.

chronyc makestep

Realiza uma atualização manual única do NTP se o chrony estiver sendo usado.

Respostas aos Exercícios Guiados

1. Insira o termo apropriado para cada definição:

Definição	Termo
Um computador que compartilha o tempo de rede com você	Provedor
A distância de um relógio de referência, em passos ou saltos	Estrato
A diferença entre o tempo do sistema e o tempo da rede	Deslocamento
A diferença entre o tempo do sistema e o tempo da rede desde a última consulta ao NTP	Variação
Grupo de servidores que fornecem tempo de rede e compartilham a carga entre si	Pool

2. Especifique os comandos usados para gerar os seguintes valores:

Valor	chronyc tracking	timedatectl show-timesync --all	ntpq -pn	chrony ntpdata	chronyc sources
Variação		X	X		
Escorregamento					
Intervalo de Consulta	X	X	X (coluna when)	X	X
Deslocamento	X		X	X	
Estrato	X	X	X	X	X
Endereço IP do provedor		X	X	X	X
Atraso da raiz	X			X	

3. Você está configurando uma rede corporativa que consiste em um servidor Linux e diversos desktops Linux. O servidor tem o endereço IP estático 192.168.0.101. Você decide que o servidor

se conectará a pool.ntp.org e, em seguida, fornecerá o tempo NTP aos computadores pessoais. Descreva a configuração do servidor e dos computadores.

Verifique se o servidor tem um serviço ntpd em execução, em vez de SNTP. Use os pools de pool.ntp.org no arquivo /etc/ntp.conf ou /etc/chrony.conf. Para cada cliente, especifique 192.168.0.101 em cada arquivo de /etc/ntp.conf ou /etc/chrony.conf.

4. Uma máquina Linux está com a hora incorreta. Descreva as etapas que você executaria para solucionar os problemas de NTP.

Primeiro, verifique se a máquina está conectada à Internet. Use ping para isso. Confira se um serviço ntpd ou SNTP está sendo executado usando systemctl status ntpd ou systemctl status systemd-timesyncd. Podem surgir mensagens de erro com informações úteis. Finalmente, use um comando como ntpq -p ou chrony tracking para verificar se alguma solicitação foi feita. Se a hora do sistema for drasticamente diferente da hora da rede, pode ser que a hora do sistema seja considerada “insana” e não possa ser alterada sem intervenção manual. Nesse caso, use um comando da lição anterior ou um comando como ntpdate pool.ntp.org para realizar uma sincronização ntp única.

Answers to Explorational Exercises

1. Pesquise as diferenças entre SNTP e NTP.

SNTP	NTP
menos preciso	mais preciso
exige menos recursos	exige mais recursos
não pode atuar como provedor de tempo	pode atuar como provedor de tempo
apenas saltos de tempo	saltos ou alterações gradativas
solicita tempo de uma única fonte	pode monitorar diversos servidores NTP e usar o provedor ideal

2. Por que um administrador de sistema optaria por *não* usar pool.ntp.org?

Do site ntppool.org: Se for absolutamente essencial ter o horário correto, é melhor considerar uma alternativa. Da mesma forma, se o seu provedor de Internet oferecer um servidor de hora, é recomendável usá-lo.

3. Como um administrador de sistema escolheria participar ou contribuir de outra forma com o projeto pool.ntp.org?

De www.ntppool.org: Seu servidor deve ter um endereço IP estático e uma conexão permanente com a Internet. O endereço IP estático não deve ser alterado ou ser alterado menos de uma vez ao ano. Além disso, os requisitos de largura de banda são modestos: 384 - 512 Kbit de largura de banda. Os servidores Estrato 3 ou 4 são bem-vindos.



108.2 Log do sistema

Referência ao LPI objectivo

[LPIC-1 version 5.0, Exam 102, Objective 108.2](#)

Peso

4

Áreas chave de conhecimento

- Configuração básica do rsyslog.
- Entendimento das facilidades (facilities), prioridades (priorities) e ações padrão.
- Consultar o diário (journal) do systemd.
- Filtrar o diário (journal) do systemd utilizando critérios como data, serviço ou prioridade.
- Apagar informações antigas do diário (journal) do systemd.
- Recuperar as informações do diário (journal) do systemd a partir de um sistema em manutenção ou uma cópia do sistema de arquivos.
- Entender a interação entre o rsyslog e o systemd-journald.
- Configuração do logrotate.
- Noções do syslog e do syslog-ng.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- `/etc/rsyslog.conf`
- `/var/log/`
- `logger`
- `logrotate`
- `/etc/logrotate.conf`

- `/etc/logrotate.d/`
- `journalctl`
- `systemd-cat`
- `/etc/systemd/journald.conf`
- `/var/log/journal/`



108.2 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	108 Serviços essenciais do sistema
Objetivo:	108.2 Registro de eventos do sistema
Lição:	1 de 2

Introdução

Os *logs* (registros de eventos do sistema) são os melhores amigos do administrador do sistema. Logs são arquivos (geralmente arquivos de texto) nos quais todos os eventos do sistema e da rede são registrados em ordem cronológica a partir do momento em que o sistema é inicializado. Assim, a gama de informações que podem ser encontradas nos logs inclui virtualmente todos os aspectos do sistema: tentativas de autenticação malsucedidas, erros de programas e serviços, hosts bloqueados pelo firewall, etc. Como você pode imaginar, os logs facilitam muito a vida dos administradores de sistema quando se trata de resolução de problemas, verificação de recursos, detecção de comportamento anômalo de programas e assim por diante.

Nesta lição, discutiremos um dos recursos de log mais comuns presentes atualmente nas distribuições GNU/Linux: o `rsyslog`. Estudaremos os diferentes tipos de logs existentes, onde são armazenados, quais informações incluem e como essas informações podem ser obtidas e filtradas. Também discutiremos como os logs podem ser mantidos em servidores centralizados em redes IP, rotação de log e o buffer de anel do kernel.

Logs do sistema

No momento em que o kernel e os diferentes processos em seu sistema começam a rodar e a se comunicar uns com os outros, muitas informações são geradas na forma de mensagens que são — em sua maioria — enviadas para os logs.

Sem os logs, os administradores de sistema teriam de descascar uma floresta de abacaxis sempre que precisassem procurar um evento que aconteceu em um servidor, donde a importância de se ter uma maneira padronizada e centralizada de rastrear e controlar quaisquer eventos do sistema. Os logs são determinantes e reveladores quando se trata de solucionar problemas e manter a segurança, além de serem fontes de dados confiáveis para se compreender as estatísticas do sistema e fazer previsões de tendências.

Deixando de lado o `systemd-journald` (que discutiremos na próxima lição), os logs tradicionalmente são tratados por três serviços dedicados principais: `syslog`, `syslog-ng` (`syslog` nova geração) e `rsyslog` (“o sistema mais veloz para processamento de log”). O `rsyslog` trouxe melhorias importantes (como suporte a RELP) e se tornou a escolha mais popular hoje em dia. Esses serviços coletam mensagens de outros serviços e programas e as armazenam em arquivos de log, normalmente em `/var/log`. No entanto, alguns serviços cuidam de seus próprios logs (por exemplo, o servidor web Apache `HTTPD` ou o sistema de impressão `CUPS`). Da mesma forma, o kernel do Linux usa um buffer de anel na memória para armazenar suas mensagens de log.

NOTE RELP significa *Reliable Event Logging Protocol* (Protocolo confiável de registro de eventos em log) e estende a funcionalidade do protocolo `syslog` de maneira a assegurar a entrega de mensagens.

Como o `rsyslog` se tornou o recurso de log padrão de fato em todas as principais distros, vamos nos concentrar nele nesta lição. O `rsyslog` usa um modelo cliente-servidor. O cliente e o servidor podem existir no mesmo host ou em máquinas diferentes. As mensagens são enviadas e recebidas em um formato específico e podem ser mantidas em servidores `rsyslog` centralizados em redes IP. O daemon do `rsyslog` — `rsyslogd` — trabalha junto com o `klogd` (que gerencia as mensagens do kernel). Nas próximas seções, discutiremos o `rsyslog` e sua infraestrutura de registro de eventos.

NOTE Um daemon é um serviço executado em segundo plano. Observe o `d` final nos nomes dos daemons: `klogd` or `rsyslogd`.

Tipos de log

Como os logs são dados variáveis, costumam ser encontrados em `/var/log`. Grosso modo, podem ser classificados em *logs do sistema* e *logs de serviços ou programas*.

Vamos conhecer alguns logs do sistema e as informações que eles preservam:

/var/log/auth.log

Atividades relacionadas aos processos de autenticação: usuários registrados, informações de sudo, cron jobs, tentativas de login malsucedidas etc.

/var/log/syslog

Um arquivo centralizado para praticamente todos os logs capturados pelo `rsyslogd`. Por incluir muitas informações, os logs são distribuídos por outros arquivos de acordo com a configuração fornecida em `/etc/rsyslog.conf`.

/var/log/debug

Informações de depuração dos programas.

/var/log/kern.log

Mensagens do kernel.

/var/log/messages

Mensagens informativas que não estão relacionadas ao kernel, mas a outros serviços. É também o destino padrão do log do cliente remoto em uma implementação de servidor de log centralizado.

/var/log/daemon.log

Informações relacionadas aos daemons ou serviços em execução em segundo plano.

/var/log/mail.log

Informações relacionadas ao servidor de email, por exemplo o postfix.

/var/log/Xorg.0.log

Informações relacionadas à placa de vídeo.

/var/run/utmp e /var/log/wtmp

Logins bem-sucedidos.

/var/log/btmp

Tentativas de login malsucedidas, por exemplo ataques de força bruta via ssh.

/var/log/faillog

Tentativas de autenticação malsucedidas.

/var/log/lastlog

Data e hora dos logins recentes do usuário.

Vejamos agora alguns exemplos de logs de serviço:

/var/log/cups/

Diretório para logs do *Common Unix Printing System*. Geralmente inclui os seguintes arquivos de log padrão: `error_log`, `page_log` e `access_log`.

/var/log/apache2/ or /var/log/httpd

Diretório para logs do *Apache Web Server*. Geralmente inclui os seguintes arquivos de log padrão: `access.log`, `error_log` e `other_vhosts_access.log`.

/var/log/mysql

Diretório para logs do *MySQL Relational Database Management System*. Geralmente inclui os seguintes arquivos de log padrão: `error_log`, `mysql.log` e `mysql-slow.log`.

/var/log/samba/

Diretório para logs do protocolo *Session Message Block* (SMB). Geralmente inclui os seguintes arquivos de log padrão: `log.`, `log.nmbd` and `log.smbd`.

NOTE

O nome e o conteúdo exatos dos arquivos de log podem variar de acordo com as distribuições Linux. Existem também logs específicos para distribuições específicas como `/var/log/dpkg.log` (contendo informações relacionadas aos pacotes dpkg) no Debian GNU/Linux e seus derivados.

Lendo logs

Para ler os arquivos de log, é preciso ser o usuário root ou ter permissões de leitura para o arquivo. Podemos usar uma variedade de utilitários, como:

less ou more

Paginadores que permitem visualizar e rolar uma página por vez:

```
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 or zmore

O mesmo que `less` e `more`, mas usados para logs que foram compactados com o `gzip` (uma função comum de `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

Visualizar as últimas linhas de um arquivo (por padrão, 10 linhas). O poder de `tail` reside — em grande parte — na opção `-f`, que mostra novas linhas dinamicamente conforme elas são anexadas:

```

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

Visualizar as primeiras linhas de um arquivo (por padrão, 10 linhas):

```
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

Utilitário de filtragem que permite buscar por strings específicas:

```
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.
(...)
```

Como você deve ter notado, a saída é impressa no seguinte formato:

- Carimbo de data/hora
- Nome do host a partir do qual a mensagem se originou
- Nome do programa/serviço que gerou a mensagem
- O PID do programa que gerou a mensagem
- Descrição da ação que ocorreu

Existem alguns exemplos em que os registros não são em forma de texto, mas arquivos binários e — consequentemente — você deverá usar comandos especiais para analisá-los:

/var/log/wtmp

Use who (ou 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

Use utmpdump ou 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

Use 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

Use 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

Também existem ferramentas gráficas para ler arquivos de log, como `gnome-logs` e `KSystemLog`.

Como as mensagens são transformadas em logs

O processo a seguir ilustra como uma mensagem é gravada em um arquivo de log:

- Aplicativos, serviços e o kernel gravam mensagens em arquivos especiais (sockets e buffers de memória), por exemplo, `/dev/log` ou `/dev/kmsg`.
- O `rsyslogd` obtém as informações dos sockets ou buffers de memória.
- Dependendo das regras encontradas em `/etc/rsyslog.conf` e/ou dos arquivos em `/etc/rsyslog.d/`, o `rsyslogd` move as informações para o arquivo de log correspondente (normalmente encontrado em `/var/log`).

NOTE

Um socket é um arquivo especial usado para transferir informações entre diferentes processos. Para listar todos os sockets em seu sistema, você pode usar o comando `systemctl list-sockets --all`.

Recursos, prioridades e ações

O arquivo de configuração de `rsyslog` é `/etc/rsyslog.conf` (em algumas distribuições, também podemos encontrar arquivos de configuração em `/etc/rsyslog.d/`). Ele normalmente é dividido em três seções: MODULES, GLOBAL DIRECTIVES e RULES. Vamos dar uma olhada neles explorando o arquivo `rsyslog.conf` em nosso host GNU/Linux 10 (buster) host — para isso, usamos `sudo less /etc/rsyslog.conf`.

MODULES inclui suporte modular para registro de eventos, capacidade de mensagem e recepção de log UDP/TCP:

```
#####
```

```
##### MODULES #####
#####
module(load="imuxsock") # provides support for local system logging
module(load="imklog")   # provides kernel logging support
#module(load="immark")  # 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 permite configurar uma série de coisas, como logs e permissões de diretório de log:

```
#####
##### 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.
#
FileChooser owner root
FileChooser group adm
FileChooser createMode 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 é onde entram os *recursos*, *prioridades* e *ações*. As configurações desta seção dizem ao daemon de log para filtrar mensagens de acordo com certas regras e registrá-las no log ou enviá-las quando necessário. Para entender essas regras, devemos primeiro explicar os conceitos de recursos e prioridades do rsyslog. Cada mensagem de log recebe um número de recurso (*facility*) e uma palavra-chave, ambos associados ao subsistema interno do Linux que produz a mensagem:

Número	Palavra-chave	Descrição
0	kern	Mensagens do kernel do Linux
1	user	Mensagens no nível do usuário
2	mail	Sistema de email
3	daemon	Daemons do sistema
4	auth, authpriv	Mensagens de segurança/autorização
5	syslog	Mensagens do syslogd
6	lpr	Subsistema de impressora de linha
7	news	Subsistema de notícias da rede
8	uucp	Subsistema UUCP (Unix-to-Unix Copy Protocol)
9	cron	Daemon do relógio
10	auth, authpriv	Mensagens de segurança/autorização
11	ftp	Daemon FTP (File Transfer Protocol)
12	ntp	Daemon NTP (Network Time Protocol)
13	security	Auditoria do log
14	console	Alerta do log
15	cron	Daemon do relógio

Número	Palavra-chave	Descrição
16 - 23	local0 até local7	Uso local 0 - 7

Além disso, cada mensagem recebe um nível de *prioridade*:

Código	Gravidade	Palavra-chave	Descrição
0	Emergência	emerg, panic	O sistema está inutilizável
1	Alerta	alert	É necessário agir imediatamente
2	Crítico	crit	Condições críticas
3	Erro	err, error	Condições de erro
4	Aviso	warn, warning	Condições de aviso
5	Atenção	notice	Condição normal, mas com alguma importância
6	Informativo	info	Mensagens informativas
7	Debug	debug	Mensagens de nível de depuração

Eis um trecho do `rsyslog.conf` de nosso sistema Debian GNU/Linux 10 (buster) que inclui alguns exemplos de regras:

```
#####
#### RULES #####
#####

# First some standard log files. Log by facility.
#
auth,authpriv.*          /var/log/auth.log
*.*,auth,authpriv.none    -/var/log/syslog
#cron.*
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

```

O formato das regras é o seguinte: <facility>.<priority> <action>

O seletor <facility>.<priority> filtra as mensagens correspondentes. Os níveis de prioridade são hierarquicamente inclusivos, o que significa que o rsyslog vai procurar mensagens com a prioridade especificada e superiores. O <action> mostra qual ação tomar (para onde enviar a mensagem de log). Aqui estão alguns exemplos para maior clareza:

auth,authpriv.*	/var/log/auth.log
-----------------	-------------------

Independentemente de sua prioridade (*), todas as mensagens dos recursos auth ou authpriv serão enviadas para /var/log/auth.log.

.;auth,authpriv.none	- /var/log/syslog
------------------------	-------------------

Todas as mensagens— independentemente de sua prioridade (*)— de todos os recursos (*)— excetuando-se as de auth ou authpriv (por isso o sufixo .none)— serão gravadas em /var/log/syslog (o sinal de menos (-) antes do caminho evita gravações em disco excessivas). Note o ponto e vírgula (;) para dividir o seletor e a vírgula (,) para concatenar dois recursos na mesma regra (auth,authpriv).

mail.err	/var/log/mail.err
----------	-------------------

As mensagens do recurso mail com um nível de prioridade de error ou superior (critical, alert ou emergency) serão enviadas para /var/log/mail.err.

```
*.=debug; \
    auth,authpriv.none; \
news.none;mail.none      -/var/log/debug
```

As mensagens de todos os recursos com a prioridade debug e nenhuma outra (=) serão gravadas em /var/log/debug—excluindo-se todas as mensagens que venham dos recursos auth, authpriv, news e mail (note a sintaxe: ; \).

Entradas manuais no log do sistema: logger

O comando logger é prático para scripts do shell ou para testes. O logger anexa todas as mensagens recebidas a /var/log/syslog (ou a /var/log/messages quando o registro for feito em um servidor de log remoto centralizado, como veremos mais adiante):

```
carol@debian:~$ logger this comment goes into "/var/log/syslog"
```

Para imprimir a última linha de /var/log/syslog, use o comando tail com a opção -1:

```
root@debian:~# tail -1 /var/log/syslog
Sep 17 17:55:33 debian carol: this comment goes into /var/log/syslog
```

rsyslog como servidor central de log

Para explicar este tópico, vamos adicionar um novo host à nossa configuração. O layout é o seguinte:

Papel	Nome do host	SO	Endereço IP
Servidor de log central	suse-server	openSUSE Leap 15.1	192.168.1.6
Cliente	debian	Debian GNU/Linux 10 (buster)	192.168.1.4

Vamos começar configurando o servidor. Em primeiro lugar, verificamos se rsyslog está

instalado e funcionando:

```
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
```

O openSUSE vem com um arquivo de configuração dedicado para log remoto: `/etc/rsyslog.d/remote.conf`. Vamos habilitar o recebimento de mensagens de clientes (hosts remotos) via TCP. Precisamos descomentar as linhas que carregam o módulo e iniciar o servidor TCP na porta 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
```

Feito isso, precisamos reiniciar o serviço rsyslog e verificar se o servidor está escutando na porta 514:

```
root@suse-server:~# systemctl restart rsyslog
root@suse-server:~# netstat -nltp | 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
```

Em seguida, vamos abrir as portas no firewall e recarregar a configuração:

```
root@suse-server:~# firewall-cmd --permanent --add-port 514/tcp
success
root@suse-server:~# firewall-cmd --reload
success
```

NOTE

Com a chegada do openSUSE Leap 15.0, `firewalld` substituiu inteiramente o clássico `SuSEFirewall2`.

Modelos e condições de filtragem

Por padrão, os logs do cliente serão gravados no arquivo `/var/log/messages` do servidor—junto com os do próprio servidor. Porém, nós vamos criar um *modelo* e uma *condição de filtragem* para armazenar os logs de nosso cliente em diretórios próprios. Para isso, adicionamos o seguinte a `/etc/rsyslog.conf` (ou `/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
```

Modelo

O modelo corresponde à primeira linha e permite especificar um formato para os nomes de log usando a geração dinâmica de nomes de arquivo. Um modelo consiste em:

- Diretriz do modelo (`$template`)
- Nome do modelo (`RemoteLogs`)
- Texto do modelo (`"/var/log/remotehosts/%HOSTNAME%/%$NOW%.%syslogseverity-text%.log"`)
- Opções (opcional)

Nosso modelo se chama `RemoteLogs` e seu texto consiste em um caminho em `/var/log`. Todos os logs de nosso host remoto irão para o diretório `remotehosts`, onde um subdiretório será criado com base no nome de host da máquina (`%HOSTNAME%`). Os nomes de arquivos neste diretório consistirão na data (`%$NOW%`), na gravidade (ou seja, a prioridade) da mensagem em formato de texto (`%syslogseverity-text%`) e no sufixo `.log`. As palavras entre os símbolos de porcentagem são *propriedades* e permitem acessar o conteúdo da mensagem de log (data, prioridade, etc.). Uma mensagem de `syslog` tem uma série de propriedades bem definidas que podem ser usadas em modelos. Essas propriedades são acessadas—e podem ser modificadas—pelo chamado

substituidor de propriedades, que implica em escrevê-las entre símbolos de porcentagem.

Condição de filtragem

As duas linhas restantes correspondem à condição de filtragem e à ação que lhe é associada:

- Filtro baseado na expressão (`if $FROMHOST-IP=='192.168.1.4'`)
- Ação (`then ?RemoteLogs, & stop`)

A primeira linha verifica o endereço IP do host remoto enviando o log e — se ele for igual ao de nosso cliente Debian — aplica o modelo `RemoteLogs`. A última linha (`& stop`) garante que as mensagens não estão sendo enviadas simultaneamente para `/var/log/messages` (mas apenas para arquivos no diretório `/var/log/remotehosts`).

NOTE Para saber mais sobre modelos, propriedades e regras, você pode consultar a página de manual de `rsyslog.conf`.

Com a configuração atualizada, reiniciamos o `rsyslog` novamente e confirmamos que ainda não existe um diretório `remotehosts` em `/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
```

O servidor agora está configurado. A seguir, vamos configurar o cliente.

Novamente, precisamos verificar se `rsyslog` está instalado e em execução:

```
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
```

Em nosso ambiente de exemplo, implementamos a resolução de nomes no cliente adicionando a linha `192.168.1.6 suse-server` a `/etc/hosts`. Assim, podemos nos referir ao servidor por nome (`suse-server`) ou endereço IP (192.168.1.6).

Nosso cliente Debian não veio com um arquivo `remote.conf` em `/etc/rsyslog.d/`, então aplicaremos nossas configurações em `/etc/rsyslog.conf`. Vamos incluir a seguinte linha no final do arquivo:

```
*.* @@suse-server:514
```

Finalmente, reiniciamos `rsyslog`.

```
root@debian:~# systemctl restart rsyslog
```

Agora, voltamos à nossa máquina `suse-server` e verificamos a existência de `remotehosts` em `/var/log`:

```
root@suse-server:~# ls /var/log/remotehosts/debian/
2019-09-17.info.log  2019-09-17.notice.log
```

Já temos dois logs em `/var/log/remotehosts`, conforme descrito em nosso modelo. Para completar esta parte, executamos `tail -f 2019-09-17.notice.log` em `suse-server` enquanto enviamos um log *manualmente* de nosso cliente Debian e confirmamos que as mensagens são anexadas ao arquivo de log como esperado (a opção `-t` fornece uma etiqueta para a mensagem):

```
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
```

O mecanismo de rotação do log

Os logs são rotacionados regularmente, o que serve a dois propósitos principais:

- Evitar que arquivos de log antigos usem mais espaço em disco do que o necessário.
- Manter os registros em um tamanho gerenciável para facilitar a consulta.

O utilitário responsável pela rotação de log é `logrotate`. Dentre suas responsabilidades estão ações como mover arquivos de log para um novo nome, arquivá-los e/ou compactá-los, às vezes enviando-os por email para o administrador do sistema e, eventualmente, excluí-los quando ficam obsoletos. Existem diversas convenções para nomear esses arquivos de log rotacionados (por exemplo, adicionar um sufixo com a data ao nome do arquivo); no entanto, a prática mais comum é simplesmente adicionar um sufixo com um número inteiro:

```
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
```

Vamos agora explicar o que acontecerá na próxima rotação do log:

1. `messages.4.gz` será excluído e perdido.
2. O conteúdo de `messages.3.gz` será movido para `messages.4.gz`.
3. O conteúdo de `messages.2.gz` será movido para `messages.3.gz`.
4. O conteúdo de `messages.1` será movido para `messages.2.gz`.
5. O conteúdo de `messages` será movido para `messages.1` e `messages` estará vazio e pronto para registrar novas entradas de log.

Observe como — de acordo com as diretrizes de `logrotate` que veremos em breve — os três arquivos de log mais antigos são compactados, mas os dois mais recentes não. Além disso, preservamos os registros das últimas 4-5 semanas. Para ler mensagens com 1 semana de idade, consultamos `messages.1` (e assim por diante).

O `logrotate` é executado diariamente como um processo automatizado ou cron job por meio do

script `/etc/cron.daily/logrotate`. Ele consulta o arquivo de configuração `/etc/logrotate.conf`. Este arquivo inclui algumas opções globais e é bem comentado; cada opção é apresentada por uma breve explicação de sua finalidade:

```
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

(...)
```

Como vemos, os arquivos de configuração em `/etc/logrotate.d` para pacotes específicos também estão incluídos. Esses arquivos contêm — na maioria — definições locais e especificam os arquivos de log a serem rotacionados (lembre-se, as definições locais têm precedência sobre as globais e as definições mais novas substituem as mais antigas). O que se segue é um trecho de uma definição em `/etc/logrotate.d/rsyslog`:

```
/var/log/messages
{
    rotate 4
    weekly
    missingok
    notifempty
    compress
    delaycompress
    sharedscripts
    postrotate
        invoke-rc.d rsyslog rotate > /dev/null
    endscript
```

{}

Como vemos, cada diretriz é separada de seu valor por um espaço em branco e/ou um sinal de igual opcional (=). As linhas entre `postrotate` e `endscript`, porém, devem aparecer em linhas próprias. A explicação é a seguinte:

rotate 4

Preserva 4 semanas de logs.

weekly

Rotaciona arquivos de log semanalmente.

missingok

Não emite uma mensagem de erro se o arquivo de log estiver ausente; simplesmente passa para o seguinte.

notifempty

Não rotaciona o log se estiver vazio.

compress

Compacta arquivos de log com o `gzip` (padrão).

delaycompress

Adia a compactação do arquivo de log anterior para o próximo ciclo de rotação (válido apenas quando usado em combinação com `compress`). Útil quando um programa não pode ser instruído a fechar seu arquivo de log e, portanto, pode continuar gravando no arquivo de log anterior por algum tempo.

sharedscripts

Relacionado aos scripts `prerotate` e `postrotate`. Para evitar que um script seja executado várias vezes, esse comando executa os scripts apenas uma vez, independentemente de quantos arquivos de log correspondem a um determinado padrão (por exemplo, `/var/log/mail/*`). Porém, os scripts não serão executados se nenhum dos logs no padrão requerer a rotação. Além disso, se os scripts forem encerrados com erros, as ações restantes não serão executadas em nenhum log.

postrotate

Indica o início de um script `postrotate`.

invoke-rc.d rsyslog rotate > /dev/null

Usa `/bin/sh` para executar `invoke-rc.d rsyslog rotate > /dev/null` depois de rotacionar os logs.

endscript

Indica o fim do script *postrotate*.

NOTE

Para uma lista completa de diretrizes e explicações, consulte a página de manual de `logrotate.conf`.

O buffer de anel do kernel

Uma vez que o kernel gera diversas mensagens antes de `rsyslogd` se tornar disponível na inicialização, torna-se necessário um mecanismo para registrar essas mensagens. É aqui que o *buffer de anel do kernel* entra em ação. Trata-se de uma estrutura de dados de tamanho fixo e — portanto — à medida que novas mensagens são gravadas, as mais antigas vão desaparecendo.

O comando `dmesg` imprime o buffer de anel do kernel. Devido ao tamanho do buffer, este comando é normalmente usado em combinação com o utilitário de filtragem de texto `grep`. Por exemplo, para pesquisar mensagens relacionadas a dispositivos Universal Serial Bus:

```
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
(...)
```

Exercícios Guiados

1. Quais utilitários/comandos você usaria nos seguintes contextos:

Finalidade e arquivo de log	Utilitário
Ler <code>/var/log/syslog.7.gz</code>	
Ler <code>/var/log/syslog</code>	
Filtrar pela palavra <code>renewal</code> em <code>/var/log/syslog</code>	
Ler <code>/var/log/faillog</code>	
Ler <code>/var/log/syslog</code> dinamicamente	

2. Reorganize as seguintes entradas de registro de forma que representem uma mensagem de registro válida com a estrutura apropriada:

- `debian-server`
- `sshd`
- `[515]:`
- `Sep 13 21:47:56`
- `Server listening on 0.0.0.0 port 22`

A ordem correta é:

3. Quais regras você adicionaria a `/etc/rsyslog.conf` para realizar as seguintes tarefas:

- Enviar todas as mensagens do recurso `mail` e uma prioridade/gravidade de `crit` (e acima) para `/var/log/mail.crit`:

- Enviar todas as mensagens do recurso `mail` com prioridades de `alert` e `emergency` para `/var/log/mail.urgent`:

- Excetuando-se as mensagens vindas dos recursos `cron` e `ntp`, enviar todas as mensagens— independentemente de seu recurso e prioridade— para `/var/log/allmessages`:

- Com todas as configurações necessárias feitas apropriadamente, enviar todas as mensagens do recurso mail para um host remoto cujo endereço IP é 192.168.1.88 usando TCP e especificando a porta padrão:

- Independentemente do recurso, enviar todas as mensagens com a prioridade warning (*somente com a prioridade warning*) para /var/log/warnings, evitando gravações excessivas no disco:

- Considere a estrofe a seguir de /etc/logrotate.d/samba e explique as diferentes opções:

```
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
}
```

Opção	Significado
weekly	
missingok	
rotate 7	
postrotate	
endscript	
compress	
delaycompress	
notifempty	

Exercícios Exploratórios

1. Na seção “Modelos e condições de filtragem”, usamos um *filtro baseado em expressão* como condição de filtragem. Os *filtros baseados em propriedades* são outro tipo de filtro exclusivo ao `rsyslogd`. Traduza nosso *filtro baseado em expressão* como um *filtro baseado em propriedades*:

Filtro baseado em expressão	Filtro baseado em propriedades
<code>if \$FROMHOST-IP=='192.168.1.4' then ?RemoteLogs</code>	

2. `omusrmsg` é um módulo interno do `rsyslog` que facilita a notificação dos usuários (ele envia mensagens de log para o terminal do usuário). Escreva uma regra para enviar todas as mensagens de *emergência* de todas as instalações para `root` e para o usuário comum `carol`.

Resumo

Nesta lição, você aprendeu:

- O registro de eventos é essencial para a administração do sistema.
- `rsyslogd` é o utilitário encarregado de manter os logs limpos e organizados.
- Alguns serviços cuidam de seus próprios logs.
- Grosso modo, os logs podem ser classificados em logs do sistema e logs de serviços/programas.
- Existem vários utilitários convenientes para a leitura de logs: `less`, `more`, `zless`, `zmore`, `grep`, `head` e `tail`.
- A maioria dos arquivos de log são arquivos de texto simples; no entanto, existe um pequeno número de arquivos de log binários.
- Em relação aos logs, o `rsyslogd` recebe as informações relevantes de arquivos especiais (sockets e buffers de memória) antes de processá-las.
- Para classificar os logs, o `rsyslogd` usa as regras de `/etc/rsyslog.conf` ou `/etc/rsyslog.d/*`.
- Qualquer usuário pode inserir suas próprias mensagens no log do sistema manualmente com o utilitário `logger`.
- O `rsyslog` permite manter todos os logs de redes IP em um servidor de log centralizado.
- Os modelos são úteis para formatar os nomes de arquivos de log de forma dinâmica.
- A finalidade da rotação de log é dupla: evitar que logs antigos ocupem espaço excessivo em disco e facilitar a consulta de logs.

Respostas aos Exercícios Guiados

1. Quais utilitários/comandos você usaria nos seguintes contextos:

Finalidade e arquivo de log	Utilitário
Ler <code>/var/log/syslog.7.gz</code>	<code>zmore</code> ou <code>zless</code>
Ler <code>/var/log/syslog</code>	<code>more</code> ou <code>less</code>
Filtrar pela palavra <code>renewal</code> em <code>/var/log/syslog</code>	<code>grep</code>
Ler <code>/var/log/faillog</code>	<code>faillog -a</code>
Ler <code>/var/log/syslog</code> dinamicamente	<code>tail -f</code>

2. Reorganize as seguintes entradas de registro de forma que representem uma mensagem de registro válida com a estrutura apropriada:

- `debian-server`
- `sshd`
- `[515]:`
- `Sep 13 21:47:56`
- `Server listening on 0.0.0.0 port 22`

A ordem correta é:

```
Sep 13 21:47:56 debian-server sshd[515]: Server listening on 0.0.0.0 port 22
```

3. Quais regras você adicionaria a `/etc/rsyslog.conf` para realizar as seguintes tarefas:

- Enviar todas as mensagens do recurso `mail` e uma prioridade/gravidade de `crit` (e acima) para `/var/log/mail.crit`:

<code>mail.crit</code>	<code>/var/log/mail.crit</code>
------------------------	---------------------------------

- Enviar todas as mensagens do recurso `mail` com prioridades de `alert` e `emergency` para `/var/log/mail.urgent`:

<code>mail.alert</code>	<code>/var/log/mail.urgent</code>
-------------------------	-----------------------------------

- Excetuando-se as mensagens vindas dos recursos `cron` e `ntp`, enviar todas as mensagens— independentemente de seu recurso e prioridade— para `/var/log/allmessages`:

```
*.*;cron.none;ntp.none           /var/log/allmessages
```

- Com todas as configurações necessárias feitas apropriadamente, enviar todas as mensagens do recurso `mail` para um host remoto cujo endereço IP é `192.168.1.88` usando TCP e especificando a porta padrão:

```
mail.* @@192.168.1.88:514
```

- Independentemente do recurso, enviar todas as mensagens com a prioridade `warning` (*somente com a prioridade warning*) para `/var/log/warnings`, evitando gravações excessivas no disco:

```
*.=warning                   -/var/log/warnings
```

4. Considere a estrofe a seguir de `/etc/logrotate.d/samba` e explique as diferentes opções:

```
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
}
```

Opção	Significado
<code>weekly</code>	Rotaciona os arquivos de log semanalmente.
<code>missingok</code>	Não emite uma mensagem de erro se o log estiver ausente; apenas passa para o seguinte.

Opção	Significado
<code>rotate 7</code>	Preserva 7 semanas de logs antigos.
<code>postrotate</code>	Executa o script na linha seguinte após rotacionar os logs.
<code>endscript</code>	Indica o fim do script <code>postrotate</code> .
<code>compress</code>	Compacta os logs com gzip.
<code>delaycompress</code>	Combinado a <code>compress</code> , adia a compactação para o ciclo de rotação seguinte.
<code>notifyempty</code>	Não rotaciona o log se ele estiver vazio.

```
[[sec.108.2_01-AEE]]
<<<
== Respostas aos Exercícios Exploratório
```

5. Na seção “Modelos e condições de filtragem”, usamos um *filtro baseado em expressão* como condição de filtragem. Os *filtros baseados em propriedades* são outro tipo de filtro exclusivo ao rsyslogd. Traduza nosso *filtro baseado em expressão* como um *filtro baseado em propriedades*:

Filtro baseado em expressão	Filtro baseado em propriedades
<code>if \$FROMHOST-IP=='192.168.1.4' then ?RemoteLogs</code>	<code>:fromhost-ip, isequal, "192.168.1.4" ?RemoteLogs</code>

6. omusrmmsg é um módulo interno do rsyslog que facilita a notificação dos usuários (ele envia mensagens de log para o terminal do usuário). Escreva uma regra para enviar todas as mensagens de *emergência* de todas as instalações para `root` e para o usuário comum `carol`.

```
*. emerg :omusrmmsg:root,carol
```



108.2 Lição 2

Certificação:	LPIC-1
Versão:	5.0
Tópico:	108 Serviços essenciais do sistema
Objetivo:	108.2 Registro de eventos do sistema
Lição:	2 de 2

Introdução

Com a adoção geral do `systemd` por todas as principais distribuições, o daemon de diário (`systemd-journald`) tornou-se o serviço de log padrão. Nesta lição, discutiremos como ele opera e como usá-lo para uma série de aplicações: consultá-lo, filtrar suas informações segundo diferentes critérios, configurar seu armazenamento e tamanho, excluir dados antigos, recuperar seus dados de um sistema de resgate ou cópia do sistema de arquivos e—por último mas não menos importante—entender sua interação com o `rsyslogd`.

Fundamentos do `systemd`

Introduzido no Fedora, o `systemd` substituiu progressivamente o SysV Init como o principal gerenciador de sistema e serviços na maioria das grandes distribuições Linux. Dentre seus pontos fortes estão os seguintes:

- Facilidade de configuração: arquivos de unidade em vez de scripts SysV Init.
- Gerenciamento versátil: além de daemons e processos, ele também gerencia dispositivos, sockets e pontos de montagem.

- Compatibilidade reversa com SysV Init e Upstart.
- Carregamento paralelo durante a inicialização: os serviços são carregados em paralelo, ao contrário do Sysv Init, que os carrega sequencialmente.
- Possui um serviço de registro denominado *diário* (journal) que apresenta as seguintes vantagens:
 - Centraliza todos os logs em um só lugar.
 - Não requer rotação de logs.
 - Os logs podem ser desabilitados, carregados na RAM ou tornados persistentes.

Unidades e destinos

O `systemd` funciona com *unidades*. Uma unidade é um recurso que o `systemd` é capaz de gerenciar (p. ex. rede, bluetooth, etc.). As unidades, por sua vez, são governadas por *arquivos de unidade*. Trata-se de arquivos de texto simples que residem em `/lib/systemd/system` e incluem as definições de configuração — na forma de *sessões* e *diretivas* — para o gerenciamento de um recurso em particular. Existem diversos tipos de unidade: `service`, `mount`, `automount`, `swap`, `timer`, `device`, `socket`, `path`, `timer`, `snapshot`, `slice`, `scope` e `target`. Assim, o nome de arquivo de cada unidade segue o padrão `<nome_do_recurso>.<tipo_de_unidade>` (p. ex. `reboot.service`).

Um *destino* é um tipo especial de unidade que se assemelha aos níveis de execução clássicos do SysV Init. Isso ocorre porque uma *unidade de destino* reúne vários recursos que representam um estado particular do sistema (por exemplo, `graphical.target` é semelhante a `runlevel 5`, etc.). Para verificar o destino atual de seu sistema, use o comando `systemctl get-default`:

```
carol@debian:~$ systemctl get-default
graphical.target
```

Por outro lado, os destinos e os níveis de execução diferem porque os primeiros são mutuamente inclusivos, enquanto os últimos não. Assim, um destino pode trazer outros destinos — o que não é possível com os níveis de execução.

NOTE Esta lição não explicará o funcionamento das unidades do `systemd`.

O diário do sistema: `systemd-journald`

`systemd-journald` é o serviço do sistema que se encarrega de receber as informações de log de uma variedade de fontes: mensagens do kernel, mensagens do sistema simples e estruturadas,

saída padrão e erro padrão dos serviços, bem como registros de auditoria do subsistema de auditoria do kernel (para saber mais, consulte a página de manual de `systemd-journald`). Sua missão é criar e manter um diário estruturado e indexado.

Seu arquivo de configuração é `/etc/systemd/journald.conf` e—como no caso de qualquer outro serviço—usamos o comando `systemctl` para *iniciá-lo*, *reiniciá-lo*, *interrompê-lo* ou—simplesmente—conferir seu *status*:

```
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
(...)
```

Arquivos de configuração do tipo `journald.conf.d/*.conf`—que podem incluir configurações específicas ao pacote—também são possíveis (consulte a página de manual de `journald.conf` para saber mais).

Quando habilitado, o diário pode ser armazenado persistentemente no disco ou de maneira volátil em um sistema de arquivos baseado na RAM. O diário não é um arquivo de texto simples, mas sim binário. Portanto, não é possível usar ferramentas de análise de texto, como `less` ou `more`, para ler seu conteúdo; o comando `journalctl` é usado em seu lugar.

Consultando o conteúdo do diário

`journalctl` é o utilitário usado para consultar o diário do `systemd`. É preciso ser root ou usar `sudo` para invocá-lo. Se consultado sem opções, ele imprimirá todo o diário em ordem cronológica (com as entradas mais antigas listadas primeiro):

```
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
(...)
```

Para fazer consultas mais específicas, há uma série de opções:

-r

As mensagens do diário serão exibidas em ordem inversa:

```
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

Imprime as mensagens mais recentes do diário e continua a imprimir as novas mensagens conforme são anexadas ao diário — semelhante a `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

Salta para o final do diário, deixando as entradas mais recentes visíveis no paginador:

```
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>

Imprime as linhas mais recentes de *value* (se não for especificado um <value>, o padrão é 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

Equivale a usar o comando 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'
(...)
```

Navegando e pesquisando no diário

Para navegar pela saída do diário, usamos:

- Teclas PageUp, PageDown e as setinhas para nos mover para cima, para baixo, a direita e a esquerda.
- `>` para ir ao final da saída.
- `<` para ir ao início da saída.

Você pode pesquisar strings para a frente e para trás a partir de sua posição atual:

- Busca para a frente: Pressione `/` e insira a string a pesquisar, depois dê Enter.
- Busca para trás: Pressione `?` e insira a string a pesquisar, depois dê Enter.

Para navegar pelas correspondências nas pesquisas, use `N` para ir para a próxima ocorrência e `Shift + N` para ir para a anterior.

Filtrando os dados do diário

O diário permite filtrar os dados de log segundo diferentes critérios:

Número de inicialização

`--list-boots`

Lista todas as inicializações disponíveis. A saída consiste em três colunas; a primeira especifica o número do boot (`0` refere-se ao boot atual, `-1` ao anterior, `-2` ao anterior ao anterior e assim por diante); a segunda coluna é o ID de inicialização; a terceira mostra as marcas temporais (timestamps):

```
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`

Mostra todas as mensagens da inicialização atual. Para ver as mensagens de log das inicializações anteriores, basta adicionar um parâmetro de deslocamento conforme explicado acima. Por exemplo, para imprimir as mensagens da inicialização anterior, digitamos `journalctl -b -1`. Lembre-se, porém, de que para recuperar informações de logs anteriores, a persistência do diário deve ser habilitada (você aprenderá como fazer isso na próxima seção):

```
root@debian:~# journalctl -b -1
Specifying boot ID has no effect, no persistent journal was found
```

Prioridade

-p

Curiosamente, também podemos filtrar por gravidade/prioridade com a opção **-p**:

```
root@debian:~# journalctl -b -0 -p err
-- No entries --
```

O diário nos informa de que—até o momento—não houve nenhuma mensagem com prioridade de **error** (ou superior) da inicialização atual. Nota: **-b -0** pode ser omitido quando nos referimos à inicialização atual.

NOTE

Consulte a lição anterior para ver uma lista completa de todas as gravidades (ou seja, prioridades) de **syslog**.

Intervalo de tempo

Podemos fazer com que o **journalctl** imprima apenas as mensagens registradas em um período de tempo específico usando as opções **--since** e **--until**. A especificação da data deve seguir o formato **AAAA-MM-DD HH:MM:SS**. A meia-noite será usada se omitirmos o componente de hora. Da mesma forma, se a data for omitida, o dia atual será pressuposto. Por exemplo, para ver as mensagens registradas das 19h às 19h01, digitamos:

```
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.
```

Da mesma maneira, podemos usar uma especificação de tempo ligeiramente diferente: "**integer time-unit ago**". Assim, para ver mensagens registradas há dois minutos, digitamos **sudo journalctl --since "2 minutes ago"**. Também é possível usar **+ e -** para especificar momentos relativos à hora atual. Assim, **--since "-2 minutes"** e **--since "2 minutes ago"** são equivalentes.

Além das expressões numéricas, podemos especificar uma série de palavras-chave:

yesterday

A partir de meia-noite do dia anterior ao dia atual.

today

A partir de meia-noite do dia atual.

tomorrow

A partir de meia-noite do dia seguinte ao dia atual.

now

A hora atual.

Vamos ver todas as mensagens desde a meia-noite passada até hoje às 21:00:

```
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

Para saber mais sobre as diferentes sintaxes para especificações de tempo, consulte a página de manual de `systemd.time`.

Programa

Para ver as mensagens do diário relacionadas a um executável específico, a seguinte sintaxe é usada: `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)
```

Unidade

Lembre-se, uma unidade é qualquer recurso gerenciado pelo `systemd` e também podemos

filtrar por elas.

-u

Mostra mensagens sobre uma unidade específica:

```
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 Para imprimir todas as unidades carregadas e ativas, use `systemctl list-units`; para ver todos os arquivos de unidade instalados, use `systemctl list-unit-files`.

Campos

O diário também pode ser filtrado por *campos* específicos através de uma das seguintes sintaxes:

- `<field-name>=<value>`
- `_<field-name>=<value>_`
- `__<field-name>=<value>`

PRIORITY=

Um dos oito valores possíveis de prioridade de `syslog` formatado como uma string decimal:

```
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
```

Note que a mesma saída poderia ser obtida com o comando `sudo journalctl -p err` visto acima.

SYSLOG_FACILITY=

Qualquer um dos números de código dos recursos formatado como uma string decimal.

Por exemplo, para ver todas as mensagens em nível de usuário:

```
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=

Mostra as mensagens produzidas por um ID de processo específico. Para ver todas as mensagens produzidas pelo `systemd`, digitariammos:

```
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

Com base no ID de inicialização, podemos selecionar as mensagens de uma inicialização específica, por exemplo: `sudo journalctl _BOOT_ID=83df3e8653474ea5aed19b41cdb45b78`.

TRANSPORT

Mostra mensagens recebidas de um transporte específico. Os valores possíveis são: `audit` (subsistema de auditoria do kernel), `driver` (gerado internamente), `syslog` (socket do `syslog`), `journal` (protocolo de diário nativo), `stdout` (saída padrão ou erro padrão dos serviços), `kernel` (buffer de anel do kernel—o mesmo que `dmesg`, `journalctl -k` ou `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...
(...)
```

Combinando campos

Os campos não são mutuamente exclusivos, portanto podemos usar mais de um na mesma consulta. No entanto, apenas as mensagens que correspondem ao valor de ambos os campos simultaneamente serão mostradas:

```
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 (...)
```

A menos que usemos o separador `+` para combinar duas expressões na forma de um *OR* lógico:

```
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= (...)
```

Por outro lado, podemos fornecer dois valores para o mesmo campo e todas as entradas correspondentes a qualquer um dos valores serão mostradas:

```
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 command[459]: __connman_inet_get_pnp_nameservers: Cannot read /pro
```

```
Oct 13 17:16:27 debian command[459]: The name net.connman.vpn was not provided by any .se
```

NOTE

Os campos do diário se enquadram em uma destas categorias: “User Journal Fields”, “Trusted Journal Fields”, “Kernel Journal Fields”, “Fields on behalf of a different program” e “Address Fields”. Para saber mais sobre esse assunto—incluindo uma lista completa dos campos—consulte a página `man` de `systemd.journal-fields(7)`.

Entradas manuais no diário do sistema: `systemd-cat`

Assim como o comando `logger` é usado para enviar mensagens da linha de comando para o log do sistema (como vimos na lição anterior), o comando `systemd-cat` tem uma finalidade semelhante—mas mais completa—with o diário do sistema. Ele nos permite enviar a entrada padrão (`stdin`), a saída (`stdout`) e o erro (`stderr`) para o diário.

Se chamado sem parâmetros, ele enviará tudo o que for lido de `stdin` para o diário. Quando terminar, pressione `Ctrl + C`:

```
carol@debian:~$ systemd-cat
This line goes into the journal.
^C
```

Se passarmos para ele a saída de um comando com pipe, ela também será enviada ao diário:

```
carol@debian:~$ echo "And so does this line." | systemd-cat
```

Se seguido por um comando, a saída desse comando também será enviada para o diário—junto com `stderr` (se houver):

```
carol@debian:~$ systemd-cat echo "And so does this line too."
```

Também existe a possibilidade de especificar um nível de prioridade com a opção `-p`:

```
carol@debian:~$ systemd-cat -p emerg echo "This is not a real emergency."
```

Consulte a página de manual de `systemd-cat` para aprender sobre as outras opções.

Para ver as últimas quatro linhas do diário:

```
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

As entradas de diário com um nível de prioridade de *emergência* serão impressas em vermelho e negrito na maioria dos sistemas.

Armazenamento persistente do diário

Como mencionado anteriormente, temos três opções quando se trata da localização do diário:

- Desativar totalmente registro em diário (porém, o redirecionamento para outros recursos, como o console, ainda será possível).
- Mantê-lo na memória—o que o torna volátil—e remover os logs a cada reinicialização do sistema. Nesse caso, o diretório `/run/log/journal` será criado e usado.
- Torná-lo persistente, escrevendo os logs no disco. Nesse caso, as mensagens de log irão para o diretório `/var/log/journal`.

O comportamento padrão é o seguinte: se `/var/log/journal/` não existir, os logs serão salvos de forma volátil em um diretório em `/run/log/journal/` e—portanto—perdidos na reinicialização. O nome do diretório—o `/etc/machine-id`—é uma string terminada em nova linha, hexadecimal, de 32 caracteres, em minúsculas:

```
carol@debian:~$ ls /run/log/journal/8821e1fdf176445697223244d1dfbd73/
system.journal
```

Se você tentar lê-lo com `less`, aparecerá um aviso, então use em vez disso o comando `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.
(....)
```

Se `/var/log/journal/` existir, os logs serão armazenados lá de maneira persistente. Caso esse diretório seja excluído, o `systemd-journald` não o recria, mas passa a escrever em `/run/log/journal`. Logo que criarmos `/var/log/journal/` novamente e reiniciarmos o daemon, o registro persistente de eventos é restabelecido:

```
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

Por padrão, há arquivos de diário específicos para todo usuário conectado, localizados em `/var/log/journal/`, por isso—junto com os arquivos de `system.journal`—também encontraremos arquivos do tipo `user-1000.journal`.

Além do que acabamos de mencionar, a forma como o daemon de diário lida com o armazenamento de logs pode ser alterada após a instalação ajustando-se seu arquivo de configuração: `/etc/systemd/journald.conf`. A opção a se usar é `Storage=`, que pode ter os seguintes valores:

`Storage=volatile`

Os dados de log serão armazenados exclusivamente na memória—em `/run/log/journal/`. Se não estiver presente, o diretório será criado.

`Storage=persistent`

Os dados de log serão armazenados em disco por padrão —em `/var/log/journal/`— com uma alternativa de segurança na memória (`/run/log/journal/`) durante os estágios iniciais da inicialização e se o disco não for gravável. Ambos os diretórios serão criados, se necessário.

Storage=auto

`auto` é semelhante a `persistent`, mas o diretório `/var/log/journal` não é criado se necessário. Este é o padrão.

Storage=none

Todos os dados de log serão descartados. Entretanto, o encaminhamento para outros destinos, como o console, o buffer de log do kernel ou um socket syslog, ainda é possível.

Por exemplo, para que `systemd-journald` crie `/var/log/journal/` e passe para o armazenamento persistente, editaríamos `/etc/systemd/journald.conf` definindo `Storage=persistent`, salváramos o arquivo e reiniciáramos o daemon com `sudo systemctl restart systemd-journald`. Para verificar se a reinicialização ocorreu sem erros, sempre podemos conferir o status do daemon:

```
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

Os arquivos de diário em `/var/log/journal/<machine-id>/` ou `/run/log/journal/<machine-id>/` têm o sufixo `.journal` (p. ex. `system.journal`). Entretanto, se estiverem corrompidos ou se o daemon for interrompido de maneira brusca, eles serão renomeados com a adição de um `~` (ou seja `system.journal~`) e o daemon passará a escrever em um novo arquivo em branco.

Excluindo dados antigos do diário: Tamanho do diário

Os logs são salvos em *arquivos de diário* com nomes de arquivos que terminam com `.journal` ou `.journal~` e estão localizados no diretório apropriado (`/run/log/journal` ou `/var/log/journal`, conforme a configuração). Para conferir quanto espaço em disco está sendo ocupado atualmente com arquivos de diário (tanto arquivados quanto ativos), use a opção `--disk-usage`:

```
root@debian:~# journalctl --disk-usage
Archived and active journals take up 24.0M in the filesystem.
```

Os logs do `systemd` têm como padrão um máximo de 10% do tamanho do sistema de arquivos onde estão armazenados. Por exemplo, em um sistema de arquivos de 1 GB, eles não ocuparão mais do que 100 MB. Assim que esse limite for atingido, os logs antigos começarão a desaparecer para se aproximar desse valor.

Porém, a aplicação de um limite de tamanho aos arquivos de diário armazenados pode ser gerenciada ajustando-se uma série de opções de configuração em `/etc/systemd/journald.conf`. Essas opções se enquadram em duas categorias, dependendo do tipo de sistema de arquivos usado: persistente (`/var/log/journal`) ou na memória (`/run/log/journal`). O primeiro usa opções prefixadas com a palavra `System` e só se aplicam se o log persistente estiver devidamente habilitado e quando o sistema estiver totalmente inicializado. Os nomes das opções do último começam com a palavra `Runtime` e se aplicarão nos seguintes cenários:

SystemMaxUse=, RuntimeMaxUse=

Controlam a quantidade de espaço em disco que pode ser ocupada pelo diário. O padrão é 10% do tamanho do sistema de arquivos, mas pode ser modificado (por exemplo, `SystemMaxUse=500M`), desde que não ultrapasse um máximo de 4GiB.

SystemKeepFree=, RuntimeKeepFree=

Controlam a quantidade de espaço em disco que deve ser deixado livre para outros usuários. O padrão é 15% do tamanho do sistema de arquivos, mas pode ser modificado (por exemplo, `SystemKeepFree = 500M`), desde que não ultrapasse um máximo de 4GiB.

Quanto à prioridade de `*MaxUse` ou `*KeepFree`, o `systemd-journald` satisfaz a ambos usando o menor dos dois valores. Da mesma forma, tenha em mente que apenas os arquivos de diário arquivados (em oposição aos ativos) são excluídos.

SystemMaxFileSize=, RuntimeMaxFileSize=

Controlam o tamanho máximo para o qual os arquivos de diário individuais podem crescer. O padrão é 1/8 de *MaxUse. A redução de tamanho é realizada de forma síncrona e os valores podem ser especificados em bytes ou usando K, M, G, T, P, E para Kibibytes, Mebibytes, Gibibyte, Tebibytes, Pebibytes e Exbibytes, respectivamente.

SystemMaxFiles=, RuntimeMaxFiles=

Estabelecem o número máximo de arquivos de diário individuais e arquivados a armazenar (os arquivos de diário ativos não são afetados). O padrão é 100.

Além da exclusão baseada no tamanho e rotação de mensagens de log, o `systemd-journald` também permite critérios baseados no tempo usando estas duas opções: `MaxRetentionSec=` e `MaxFileSec=`. Consulte a página de manual do `journald.conf` para saber mais sobre estas e outras opções.

NOTE Sempre que você modificar o comportamento padrão do `systemd-journald` descomentando e editando as opções em `/etc/systemd/journald.conf`, será preciso reiniciar o daemon para que as mudanças tenham efeito.

Limpando o diário

É possível limpar (vacuum) manualmente os arquivos de diário arquivados a qualquer momento com uma das três opções a seguir:

--vacuum-time=

Esta opção baseada remove todas as mensagens nos arquivos de diário que tiverem um carimbo de data/hora anterior ao período de tempo especificado. Os valores devem ser escritos com qualquer um dos seguintes sufixos: s, m, h, days (ou d), months, weeks (or w) e years (ou y). Por exemplo, para se livrar de todas as mensagens com mais de 1 mês em arquivos de diário arquivados:

```
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-00000000000008e-00059479df9677c8.journal (8.0M).
```

--vacuum-size=

Esta opção remove arquivos de diário arquivados até que cheguem a um tamanho abaixo do

especificado. Os valores devem ser escritos com qualquer um dos seguintes sufixos: K, M, G ou T. Por exemplo, para eliminar arquivos de diário arquivados até que estejam abaixo de 100 Mebibytes:

```
root@debian:~# journalctl --vacuum-size=100M
Vacuuming done, freed 0B of archived journals from
/run/log/journal/9a32ba45ce44423a97d6397918de1fa5.
```

--vacuum-files=

Esta opção cuida para que não restem mais arquivos de diário arquivados do que o número especificado. O valor é um número inteiro. Por exemplo, para limitar o número de arquivos de diário arquivados a 10:

```
root@debian:~# journalctl --vacuum-files=10
Vacuuming done, freed 0B of archived journals from
/run/log/journal/9a32ba45ce44423a97d6397918de1fa5.
```

Essa limpeza remove apenas os arquivos de diário arquivados. Se a ideia for se livrar de tudo (incluindo arquivos de diário ativos), temos de empregar um sinal (SIGUSR2) que solicita a rotação imediata dos arquivos de diário com a opção **--rotate**. Outros sinais importantes podem ser chamados com as seguintes opções:

--flush (SIGUSR1)

Solicita a liberação dos arquivos de diário de `/run/` para `/var/` para tornar o diário persistente. Ele requer que o log persistente esteja habilitado e que `/var/` esteja montado.

--sync (SIGRTMIN+1)

É usado para solicitar que todos os dados de log não gravados sejam gravados no disco.

NOTE Para verificar a consistência interna do arquivo de diário, use `journalctl` com a opção **--verify**. Você verá uma barra de progresso quando a verificação for concluída e todos os problemas possíveis serão mostrados.

Recuperando dados de diário de um sistema de resgate

Como administrador de sistema, você pode se encontrar em uma situação em que precisa acessar arquivos de diário no disco rígido de uma máquina defeituosa por meio de um sistema de recuperação (um CD inicializável ou pendrive contendo uma distribuição Linux ativa).

`journalctl` procura por arquivos de diário em `/var/log/journal/<machine-id>/`. Como os

IDs de máquina dos sistemas de resgate e dos sistemas defeituosos serão diferentes, usamos a seguinte opção:

-D </path/to/dir>, --directory=</path/to/dir>

Com esta opção, especificamos um caminho de diretório no qual `journalctl` busca por arquivos de diário ao invés do tempo de execução e locais do sistema.

Portanto, é necessário montar o `rootfs` (`/dev/sda1`) do sistema defeituoso no sistema de arquivos do sistema de recuperação e em seguida ler os arquivos de diário da seguinte forma:

```
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 4.12.14-1p151.28.16-default
(geeko@buildhost) (...)
oct 20 12:30:45 suse-server kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.12.14-
1p151.28.16-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'
(...)
```

Outras opções que podem ser úteis neste caso:

-m, --merge

Mescla as entradas de todos os diários disponíveis em `/var/log/journal`, incluindo os remotos.

--file

Mostra as entradas de um arquivo específico, por exemplo: `journalctl --file /var/log/journal/64319965bda04dfa81d3bc4e7919814a/user-1000.journal`.

--root

Um caminho de diretório que indica que o diretório raiz é passado como um argumento. `journalctl` procura por arquivos de diário (por exemplo, `journalctl --root /faulty.system/`).

Veja a página de manual do `journalctl` para saber mais.

Encaminhando dados de log para um daemon syslog tradicional

Os dados de log do diário podem ser disponibilizados para um daemon `syslog` tradicional das

seguintes maneiras:

- Encaminhando mensagens para o arquivo de socket `/run/systemd/journal/syslog` para `syslogd` ler. Este recurso é habilitado com a opção `ForwardToSyslog=yes`.
- Fazendo com que um daemon `syslog` se comporte como `journalctl`, lendo assim mensagens de log diretamente nos arquivos de diário. Neste caso, a opção relevante é `Storage`; ela deve ter um valor diferente de `none`.

NOTE

Da mesma forma, é possível encaminhar mensagens de log para outros destinos com as seguintes opções: `ForwardToKMsg` (buffer de log do kernel — `kmsg`), `ForwardToConsole` (console do sistema) ou `ForwardToWall` (todos os usuários logados via `wall`). Para mais informações, consulte a página de manual do `journald.conf`.

Exercícios Guiados

1. Pressupondo que você é `root`, complete a tabela com o comando apropriado do `journalctl`:

Finalidade	Comando
Imprimir entradas do kernel	
Imprimir mensagens da segunda inicialização a partir do início do diário	
Imprimir mensagens da segunda inicialização a partir do final do diário	
Imprimir as mensagens de log mais recentes e ficar atento às novas	
Imprimir apenas novas mensagens a partir de agora e atualizar a saída continuamente	
Imprimir mensagens da inicialização anterior com prioridade de <code>warning</code> e na ordem inversa	

2. O comportamento do daemon de diário em relação ao armazenamento é controlado principalmente pelo valor da opção `Storage` em `/etc/systemd/journald.conf`. Indique qual comportamento está relacionado a qual valor na seguinte tabela:

Comportamento	<code>Storage=auto</code>	<code>Storage=none</code>	<code>Storage=persistent</code>	<code>Storage=volatile</code>
Os dados de registro são descartados, mas o encaminhamento é possível.				

Comportamento	Storage=auto	Storage=none	Storage=persistent	Storage=volatile
Assim que o sistema for inicializado, os dados de registro serão armazenados em <code>/var/log/journal</code> . Se ainda não estiver presente, o diretório será criado.				
Assim que o sistema for inicializado, os dados de registro serão armazenados em <code>/var/log/journal</code> . Se ainda não estiver presente, o diretório não será criado.				
Os dados de log serão armazenados em <code>/var/run/journal</code> , mas não sobreviverão a reinicializações.				

3. Como você aprendeu, o diário pode ser limpo manualmente com base no tempo, tamanho e número de arquivos. Conclua as seguintes tarefas usando `journalctl` e as opções apropriadas:
- Verifique quanto espaço em disco é ocupado pelos arquivos de diário:

- Reduza a quantidade de espaço reservado a arquivos de diário arquivados e defina-o como

200 MiB:

- Verifique o espaço em disco novamente e explique os resultados:

Exercícios Exploratórios

1. Quais opções você precisa modificar em `/etc/systemd/journald.conf` para que as mensagens sejam encaminhadas para `/dev/tty5`? Quais valores as opções devem ter?

2. Informe qual o filtro `journalctl` correto para imprimir o seguinte:

Finalidade	Filtro + Valor
Imprimir mensagens pertencentes a um usuário específico	
Imprimir mensagens de um host chamado <code>debian</code>	
Imprimir mensagens pertencentes a um grupo específico	
Imprimir mensagens pertencentes a <code>root</code>	
Com base no caminho do executável, imprimir mensagens <code>sudo</code>	
Com base no nome do comando, imprimir mensagens <code>sudo</code>	

3. Ao filtrar por prioridade, os registros com uma prioridade mais alta do que a indicada também serão incluídos na lista; por exemplo, `journalctl -p err` irá imprimir as mensagens `error`, `critical`, `alert` e `emergency`. No entanto, você pode fazer com que o `journalctl` mostre apenas um intervalo específico. Qual comando você usaria para fazer com que o `journalctl` imprima apenas mensagens nos níveis de prioridade `warning`, `error` e `critical`?

4. Os níveis de prioridade também podem ser especificados numericamente. Reescreva o comando do exercício anterior usando a representação numérica dos níveis de prioridade:

Summary

Nesta lição, você aprendeu:

- As vantagens de usar o `systemd` como um gerenciador de sistema e serviços.
- O básico sobre as unidades e destinos do `systemd`.
- De onde `systemd-journald` obtém os dados de log.
- As opções que podemos passar ao `systemctl` para controlar `systemd-journald`: `start`, `status`, `restart` e `stop`.
- Onde o arquivo de configuração do diário está localizado — `/etc/systemd/journald.conf` — e suas principais opções.
- Como consultar o diário de maneira geral e buscar por dados específicos usando filtros.
- Como navegar e pesquisar no diário.
- Como lidar com o armazenamento de arquivos de diário: na memória ou no disco.
- Como desativar completamente o registro no diário.
- Como verificar o espaço em disco ocupado pelo diário, impor limites de tamanho aos arquivos de diário armazenados e limpar os arquivos de diário arquivados manualmente (*vacuuming*).
- Como recuperar dados de diário de um sistema de resgate.
- Como encaminhar dados de log para um daemon `syslog` tradicional.

Comandos usados nesta lição:

systemctl

Controlar o gerenciador de sistema e serviços `systemd`.

journalctl

Consultar o diário do `systemd`.

ls

Listar o conteúdo do diretório.

less

Visualizar o conteúdo do arquivo.

mkdir

Criar diretórios.

Answers to Guided Exercises

1. Pressupondo que você é `root`, complete a tabela com o comando apropriado do `journalctl`:

Finalidade	Comando
Imprimir entradas do kernel	<code>journalctl -k or journalctl --dmesg</code>
Imprimir mensagens da segunda inicialização a partir do início do diário	<code>journalctl -b 2</code>
Imprimir mensagens da segunda inicialização a partir do final do diário	<code>journalctl -b -2 -r or journalctl -r -b -2</code>
Imprimir as mensagens de log mais recentes e ficar atento às novas	<code>journalctl -f</code>
Imprimir apenas novas mensagens a partir de agora e atualizar a saída continuamente	<code>journalctl --since "now" -f</code>
Imprimir mensagens da inicialização anterior com prioridade de <code>warning</code> e na ordem inversa	<code>journalctl -b -1 -p warning -r</code>

2. O comportamento do daemon de diário em relação ao armazenamento é controlado principalmente pelo valor da opção `Storage` em `/etc/systemd/journald.conf`. Indique qual comportamento está relacionado a qual valor na seguinte tabela:

Comportamento	<code>Storage=auto</code>	<code>Storage=none</code>	<code>Storage=persistent</code>	<code>Storage=volatile</code>
Os dados de registro são descartados, mas o encaminhamento é possível		x		

Comportamento	Storage=auto	Storage=none	Storage=persistent	Storage=volatile
Assim que o sistema for inicializado, os dados de registro serão armazenados em <code>/var/log/journal</code> . Se ainda não estiver presente, o diretório será criado			x	
Assim que o sistema for inicializado, os dados de registro serão armazenados em <code>/var/log/journal</code> . Se ainda não estiver presente, o diretório não será criado	x			
Os dados de log serão armazenados em <code>/var/run/journal</code> , mas não sobreviverão a reinicializações				x

3. Como você aprendeu, o diário pode ser limpo manualmente com base no tempo, tamanho e número de arquivos. Conclua as seguintes tarefas usando `journalctl` e as opções apropriadas:
- Verifique quanto espaço em disco é ocupado pelos arquivos de diário:

```
journalctl --disk-usage
```

- Reduza a quantidade de espaço reservado a arquivos de diário arquivados e defina-o como 200 MiB:

```
journalctl --vacuum-size=200M
```

- Verifique o espaço em disco novamente e explique os resultados:

```
journalctl --disk-usage
```

Não há correlação porque `--disk-usage` mostra o espaço ocupado pelos arquivos de diário ativos e arquivados, ao passo que `--vacuum-size` aplica-se apenas aos itens arquivados.

Answers to Explorational Exercises

1. Quais opções você precisa modificar em `/etc/systemd/journald.conf` para que as mensagens sejam encaminhadas para `/dev/tty5`? Quais valores as opções devem ter?

```
ForwardToConsole=yes
TTYPath=/dev/tty5
```

2. Informe qual o filtro `journalctl` correto para imprimir o seguinte:

Finalidade	Filtro + Valor
Imprimir mensagens pertencentes a um usuário específico	<code>_ID=<user-id></code>
Imprimir mensagens de um host chamado <code>debian</code>	<code>_HOSTNAME=debian</code>
Imprimir mensagens pertencentes a um grupo específico	<code>_GID=<group-id></code>
Imprimir mensagens pertencentes a <code>root</code>	<code>_UID=0</code>
Com base no caminho do executável, imprimir mensagens <code>sudo</code>	<code>_EXE=/usr/bin/sudo</code>
Com base no nome do comando, imprimir mensagens <code>sudo</code>	<code>_COMM=sudo</code>

3. Ao filtrar por prioridade, os registros com uma prioridade mais alta do que a indicada também serão incluídos na lista; por exemplo, `journalctl -p err` irá imprimir as mensagens `error`, `critical`, `alert` e `emergency`. No entanto, você pode fazer com que o `journalctl` mostre apenas um intervalo específico. Qual comando você usaria para fazer com que o `journalctl` imprima apenas mensagens nos níveis de prioridade `warning`, `error` e `critical`?

```
journalctl -p warning..crit
```

4. Os níveis de prioridade também podem ser especificados numericamente. Reescreva o comando do exercício anterior usando a representação numérica dos níveis de prioridade:

```
journalctl -p 4..2
```



Linux
Professional
Institute

108.3 Fundamentos de MTA (Mail Transfer Agent)

Referência ao LPI objectivo

LPIC-1 version 5.0, Exam 102, Objective 108.3

Peso

3

Áreas chave de conhecimento

- Criar aliases de e-mail.
- Configurar o redirecionamento de e-mail.
- Conhecimento sobre os programas MTA comumente usados (postfix, sendmail, qmail, exim) (não é cobrada a configuração desses programas)

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- `~/.forward`
- Comandos que simulam o sendmail
- `newaliases`
- `mail`
- `mailq`
- `postfix`
- `sendmail`
- `exim`



108.3 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	108 Serviços essenciais do sistema
Objetivo:	108.3 Noções básicas do Mail Transfer Agent (MTA)
Lição:	1 de 1

Introdução

Nos sistemas operacionais do tipo Unix, como o Linux, cada usuário tem sua própria *caixa de entrada*: um local especial no sistema de arquivos, inacessível a outros usuários não-root, para armazenar as mensagens de email pessoais do usuário. Novas mensagens são adicionadas à caixa de entrada do usuário pelo *Mail Transfer Agent* (MTA). O MTA é um programa executado como um serviço do sistema que coleta mensagens enviadas por outras contas locais, bem como mensagens recebidas da rede, enviadas por contas de usuários remotos.

O mesmo MTA é responsável pelo envio de mensagens para a rede, caso o endereço de destino se refira a uma conta remota. Ele faz isso usando um local do sistema de arquivos como uma *caixa de saída* de email para todos os usuários do sistema: assim que um usuário coloca uma nova mensagem na caixa de saída, o MTA identifica o nó da rede de destino a partir do nome de domínio fornecido pelo endereço de email do destino—a parte após o sinal @—e então tenta transferir a mensagem para o MTA remoto usando o *Simple Mail Transfer Protocol* (SMTP). O SMTP foi projetado pensando em redes não confiáveis; portanto, ele tenta estabelecer rotas de entrega alternativas caso o nó de destino principal do email esteja inacessível.

MTA local e remoto

As contas de usuário tradicionais das máquinas conectadas em rede constituem o cenário mais simples de troca de emails, em que cada nó da rede executa seu próprio daemon MTA. Nenhum outro software além do MTA é necessário para enviar e receber mensagens de email. Na prática, porém, é mais comum usar uma conta de email remota e não ter um serviço MTA local ativo (ou seja, usar um aplicativo cliente de email para acessar a conta remota).

Ao contrário das contas locais, uma conta de email remota — também chamada de *caixa de correio remota* — requer autenticação do usuário para conceder acesso à caixa de correio do usuário e ao MTA remoto (neste caso, simplesmente chamado de *servidor SMTP*). Ao passo que o usuário que interage com uma caixa de entrada local e o MTA já está identificado pelo sistema, um sistema remoto precisa verificar a identidade do usuário antes de manipular suas mensagens por meio de IMAP ou POP3.

NOTE

Hoje em dia, o método mais comum para enviar e receber emails é através de uma conta hospedada em um servidor remoto; por exemplo, um servidor de email centralizado de uma empresa que hospeda todas as contas dos funcionários ou um serviço de email pessoal, como o *Gmail* do Google. Em vez de coletar as mensagens entregues localmente, o aplicativo cliente de email se conecta à caixa de correio remota e recupera as mensagens de lá. Os protocolos POP3 e IMAP são comumente usados para recuperar as mensagens no servidor remoto, mas outros protocolos proprietários não padrão também podem ser usados.

Quando um daemon MTA está sendo executado no sistema local, os usuários locais podem enviar um email para outros usuários locais ou usuários em uma máquina remota, desde que seu sistema também tenha um serviço MTA que aceite conexões de rede. A porta TCP 25 é a porta padrão para a comunicação SMTP, mas outras portas também podem ser usadas, dependendo do esquema de autenticação e/ou criptografia empregado (se houver).

Deixando de lado as topologias que envolvem o acesso a caixas de correio remotas, uma rede de troca de emails entre contas de usuários Linux comuns pode ser implementada, desde que todos os nós da rede tenham um MTA ativo capaz de realizar as seguintes tarefas:

- Manter a fila de saída das mensagens a serem enviadas. Para cada mensagem na fila, o MTA local avaliará o MTA de destino a partir do endereço do destinatário.
- Comunicar-se com demônios MTA remotos usando SMTP. O MTA local deve ser capaz de usar o protocolo SMTP sobre a pilha TCP/IP para receber, enviar e redirecionar mensagens de/para outros demônios MTA remotos.
- Manter uma caixa de entrada individual para cada conta local. O MTA geralmente armazena as

mensagens no formato *mbox*: um único arquivo de texto contendo todas as mensagens de email em sequência.

Normalmente, os endereços de email especificam um nome de domínio como o local, por exemplo `lpi.org` em `info@lpi.org`. Quando for esse o caso, o MTA do remetente consultará o serviço de DNS em busca do registro MX correspondente. O registro DNS MX contém o endereço IP do MTA que gerencia o email para esse domínio. Se o mesmo domínio tiver mais de um registro MX especificado no DNS, o MTA tentará contatá-los de acordo com seus valores de prioridade. Se o endereço do destinatário não especificar um nome de domínio ou se o domínio não tiver um registro MX, a parte após o símbolo @ será tratada como o host do MTA de destino.

É preciso pensar na segurança se os hosts MTA forem ficar visíveis para os hosts na internet. Por exemplo, um usuário desconhecido poderia usar o MTA local para se passar por outro usuário e enviar emails potencialmente perigosos. Um MTA que retransmite cegamente um email é conhecido como *open relay*, quando pode ser usado como intermediário para potencialmente disfarçar o verdadeiro remetente da mensagem. Para evitar esses usos indevidos, recomenda-se aceitar conexões somente de domínios autorizados e implementar um esquema de autenticação seguro.

Além disso, existem muitas implementações de MTA diferentes para Linux, cada qual enfocando aspectos específicos como compatibilidade, desempenho, segurança etc. No entanto, todos os MTAs seguem os mesmos princípios básicos e fornecem recursos semelhantes.

MTAs do Linux

O MTA tradicionalmente disponível para os sistemas Linux é o *Sendmail*, um MTA de uso geral bastante flexível adotado por muitos sistemas operacionais do tipo Unix. Outros MTAs comuns são o *Postfix*, o *qmail* e o *Exim*. O principal motivo para escolher um MTA alternativo é implementar recursos avançados com mais facilidade, pois configurar servidores de email personalizados no Sendmail pode ser uma tarefa complicada. Além disso, cada distribuição pode ter seu MTA preferido, com ajustes predefinidos apropriados à maioria das configurações comuns. Todos os MTAs pretendem ser substitutos diretos do Sendmail e, portanto, todos os aplicativos compatíveis com o Sendmail costumam funcionar, independentemente do MTA que está sendo usado.

Se o MTA estiver em execução, mas não aceitar conexões de rede, ele só poderá entregar mensagens de email na máquina local. Para o MTA `sendmail`, o arquivo `/etc/mail/sendmail.mc` deve ser modificado para aceitar conexões não locais. Para isso, a entrada

```
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

deve ser modificada com o endereço de rede correto e o serviço precisa ser reiniciado. Algumas distribuições Linux, como o Debian, podem oferecer ferramentas de configuração para ajudar a garantir que o servidor de email com um conjunto predefinido de recursos comumente usados.

TIP Devido a problemas de segurança, a maioria das distribuições Linux não instala um MTA por padrão. Para testar os exemplos fornecidos nesta lição, verifique se existe um MTA em execução em todas as máquinas e se elas aceitam conexões na porta TCP 25. Para fins de segurança, esses sistemas não devem ser expostos a conexões de entrada da internet pública durante o teste.

Uma vez que o MTA está em execução e aceitando conexões da rede, as novas mensagens de email são passadas para ele com comandos SMTP enviados por meio de uma conexão TCP. O comando `nc` — um utilitário de rede que lê e grava dados genéricos na rede — pode ser usado para enviar comandos SMTP diretamente para o MTA. Se o comando `nc` não estiver disponível, ele será instalado com o pacote `ncat` ou `nmap-ncat`, dependendo do sistema de gerenciamento de pacotes em uso. Escrever comandos SMTP diretamente no MTA é útil para entender melhor o protocolo e outros conceitos gerais sobre o email, bem como para ajudar a diagnosticar problemas no processo de entrega de mensagens.

Se, por exemplo, a usuária `emma` no host `lab1.campus` deseja enviar uma mensagem ao usuário `dave` no host `lab2.campus`, ela pode usar o comando `nc` para conectar-se diretamente ao MTA `lab2.campus`, supondo que ele esteja escutando na porta 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
```

Assim que a conexão é estabelecida, o MTA remoto se identifica e está pronto para receber

comandos SMTP. O primeiro comando SMTP no exemplo, **HELO lab1.campus**, indica **lab1.campus** como o iniciador da troca. Os dois comandos seguintes, **MAIL FROM: emma@lab1.campus** e **RCPT TO: dave@lab2.campus**, indicam o remetente e o destinatário. A mensagem de email em si começa após o comando **DATA** e termina com um ponto sozinho em uma linha. Para adicionar um campo **subject** (assunto) ao email, ele deve estar na primeira linha após o comando **DATA**, como mostrado no exemplo. Quando o campo de assunto é usado, deve haver uma linha em branco separando-o do conteúdo do email. O comando **QUIT** termina a conexão com o MTA no host **lab2.campus**.

No host **lab2.campus**, o usuário **dave** receberá uma mensagem semelhante a **You have new mail in /var/spool/mail/dave** assim que entrar em uma sessão do shell. Este arquivo conterá a mensagem de email bruta enviada por **emma**, bem como os cabeçalhos adicionados pelo MTA:

```
$ 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.
```

O cabeçalho **Received:** mostra que a mensagem de **lab1.campus** foi recebida diretamente por **lab2.campus**. Por padrão, os MTAs só aceitam mensagens para destinatários locais. O seguinte erro provavelmente ocorrerá se o usuário **emma** tentar enviar um email para o usuário **henry** no host **lab3.campus** usando o MTA **lab2.campus** em vez do MTA correto **lab3.campus**:

```
$ 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
```

Os números de resposta do SMTP que começam com 5, como na mensagem `Relaying denied` (retransmissão negada), indicam um erro. Existem situações legítimas em que a retransmissão é desejável, como quando os hosts que enviam e recebem emails não estão conectados o tempo todo: um MTA intermediário pode ser configurado para aceitar emails destinados a outros hosts, agindo como um servidor SMTP *retransmissor* (ou relay) capaz de encaminhar mensagens entre MTAs.

A possibilidade de rotear o tráfego de email por meio de servidores SMTP intermediários desestimula a tentativa de conexão direta ao host indicado pelo endereço de email do destinatário, como mostrado nos exemplos anteriores. Além disso, os endereços de email geralmente têm um nome de domínio como local (após o @), de forma que o nome real do host MTA correspondente deve ser recuperado através do DNS. Portanto, é recomendável delegar a tarefa de identificar o host de destino apropriado ao MTA local ou ao servidor SMTP remoto quando usamos caixas de correio remotas.

O Sendmail fornece o comando `sendmail` para realizar uma série de operações relacionadas a email, incluindo auxiliar na composição de novas mensagens. Ele também requer que o usuário digite os cabeçalhos do email manualmente, mas de uma forma mais amigável do que usar comandos SMTP diretamente. Assim, um método mais adequado para o usuário `emma@lab1.campus` enviar uma mensagem de email para `dave@lab2.campus` seria:

```
$ 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.
.
```

Aqui, novamente, o ponto sozinho em uma linha finaliza a mensagem. A mensagem deve ser enviada imediatamente ao destinatário, a menos que o MTA local não tenha conseguido contatar o MTA remoto. O comando `mailq`, se executado pelo root, mostra todas as mensagens não entregues. Se, por exemplo, o MTA em `lab2.campus` não respondeu, então o comando `mailq` listará a mensagem não entregue e a causa da falha:

```
# 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

O local padrão da fila da caixa de saída é `/var/spool/mqueue/`, mas diferentes MTAs podem usar locais diversos no diretório `/var/spool/`. O Postfix, por exemplo, cria uma árvore de diretórios sob `/var/spool/postfix/` para gerenciar a fila. O comando `mailq` equivale a `sendmail -bp`, e eles devem estar presentes qualquer que seja o MTA instalado no sistema. Para garantir a compatibilidade reversa, a maioria dos MTAs inclui esses comandos tradicionais de administração de email.

Se o principal host de destino do email — quando fornecido a partir de um registro MX DNS para o domínio — estiver inacessível, o MTA tentará entrar em contato com as entradas com prioridade mais baixa (se houver alguma especificada). Se nenhuma delas estiver acessível, a mensagem ficará na fila da caixa de saída local para ser enviada posteriormente. Se configurado para isso, o MTA pode verificar periodicamente a disponibilidade dos hosts remotos e realizar uma nova tentativa de entrega. Se estiver usando um MTA compatível com o Sendmail, uma nova tentativa ocorrerá imediatamente com o comando `sendmail -q`.

O Sendmail armazena as mensagens recebidas em um arquivo com o nome do proprietário da caixa de entrada correspondente, por exemplo `/var/spool/mail/dave`. Outros MTAs, como o Postfix, podem armazenar as mensagens de email recebidas em locais como `/var/mail/dave`, mas o conteúdo do arquivo é o mesmo. No exemplo, o comando `sendmail` foi usado no host do remetente para compor a mensagem, de modo que os cabeçalhos brutos da mensagem mostram que o email atravessou etapas extras antes de chegar ao destino final:

```
$ 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.

De baixo para cima, as linhas que começam com Received: mostram a rota seguida pela mensagem. A mensagem foi enviada pela usuária emma com o comando sendmail dave@lab2.campus emitido em lab1.campus, conforme declarado pelo primeiro cabeçalho Received:. Em seguida, ainda em lab1.campus, o MTA usa ESMTPS—um superconjunto do SMTP, que adiciona extensões de criptografia—para enviar a mensagem para o MTA em lab2.campus, conforme especificado pelo último (topo) cabeçalho Received:.

O MTA conclui seu trabalho depois que a mensagem é salva na caixa de entrada do usuário. É comum realizar algum tipo de filtragem de email, como bloqueadores de spam ou a aplicação de regras de filtragem definidas pelo usuário. Essas tarefas são executadas por aplicativos de terceiros, trabalhando em conjunto com o MTA. O MTA poderia, por exemplo, chamar o utilitário *SpamAssassin* para marcar mensagens suspeitas usando seus recursos de análise de texto.

Embora possível, não é conveniente ler diretamente o arquivo da caixa de correio. Recomenda-se usar um programa cliente de email (por exemplo, Thunderbird, Evolution ou KMail), que analisará o arquivo e gerenciará as mensagens apropriadamente. Esses programas também oferecem recursos extras, como atalhos para ações comuns, subdiretórios de caixa de entrada, etc.

O comando mail e Mail User Agents (MUA)

É possível escrever uma mensagem de email diretamente em seu formato bruto, mas é muito mais prático usar um aplicativo cliente—também conhecido como MUA (*Mail User Agent* ou agente do usuário de email)—para agilizar o processo e evitar erros. O MUA cuida do trabalho de bastidores, ou seja, o cliente de email apresenta e organiza as mensagens recebidas e faz a comunicação adequada com o MTA após o usuário redigir um email.

Existem muitos tipos distintos de Mail User Agents. Aplicativos de desktop como o *Mozilla Thunderbird* e o *Evolution*, do Gnome, oferecem suporte a contas de email locais e remotas. Mesmo as interfaces *Webmail* podem ser vistas como um tipo de MUA, pois intermediam a interação entre o usuário e o MTA subjacente. No entanto, os clientes de email não estão restritos às interfaces gráficas: os clientes de email de console são amplamente usados para acessar caixas de correio não integradas a uma interface gráfica e para automatizar tarefas relacionadas a email em scripts do shell.

Originalmente, o comando `mail` do Unix destinava-se apenas a compartilhar mensagens entre usuários do sistema local (o primeiro comando `mail` data da primeira edição do Unix, lançada em 1971). Quando as trocas de emails em rede se tornaram mais expressivas, outros programas

foram criados para lidar com o novo sistema de entrega e gradualmente substituíram o antigo programa mail.

Hoje em dia, o comando mail mais comumente usado é fornecido pelo pacote *mailx*, compatível com todos os recursos de email modernos. Na maioria das distribuições Linux, o comando mail é apenas um link simbólico para o comando *mailx*. Outras implementações, como o pacote *GNU Mailutils*, basicamente fornecem os mesmos recursos do *mailx*. Existem, no entanto, pequenas diferenças entre eles, especialmente com relação às opções de linha de comando.

Independentemente de sua implementação, todas as variações modernas do comando mail operam em dois modos: *modo normal* e *modo de envio*. Se um endereço de email for fornecido como argumento para o comando mail, ele entrará no modo de envio; caso contrário, entrará no modo normal (leitura). No modo normal, as mensagens recebidas são listadas com um índice numérico para cada uma, para que o usuário possa consultá-las individualmente ao digitar comandos no prompt interativo. O comando print 1 pode ser usado para exibir o conteúdo da mensagem número 1, por exemplo. Comandos interativos podem ser abreviados, de forma que comandos como print, delete ou reply podem ser substituídos por p, d ou r, respectivamente. O comando mail sempre considerará a última mensagem recebida ou visualizada quando o número do índice da mensagem for omitido. O comando quit ou q serve para sair do programa.

O *modo de envio* (send mode) é especialmente útil para enviar mensagens de email automatizadas. Ele pode ser usado, por exemplo, para enviar um email ao administrador do sistema se um script de manutenção programado falhar em realizar sua tarefa. No modo de envio, o mail usa o conteúdo da *entrada padrão* como corpo da mensagem:

```
$ mail -s "Maintenance fail" henry@lab3.campus <<<"The maintenance script failed at `date`"
```

Neste exemplo, a opção -s foi adicionada para incluir um campo de assunto na mensagem. O corpo da mensagem foi fornecido pelo redirecionamento *Hereline* para a entrada padrão, mas o conteúdo de um arquivo ou a saída de um comando também pode ser canalizado com um pipe para o *stdin* do programa. Se nenhum conteúdo for fornecido por um redirecionamento para a entrada padrão, o programa aguardará que o usuário entre no corpo da mensagem. Nesse caso, a combinação de teclas **Ctrl + D** encerra a mensagem. O comando mail é concluído imediatamente após a mensagem ser adicionada à fila da caixa de saída.

Entrega personalizada

Por padrão, as contas de email em um sistema Linux são associadas às contas de sistema padrão. Por exemplo, se a usuária Carol tem o nome de login carol no host lab2.campus, seu endereço de email será carol@lab2.campus. Esta associação direta entre contas de sistema e caixas de

correio pode ser estendida por métodos padrão fornecidos pela maioria das distribuições Linux, em particular o mecanismo de roteamento de email fornecido pelo arquivo `/etc/aliases`.

Um alias de email é um destinatário de email “virtual” cujas mensagens recebidas são redirecionadas para caixas de correio locais existentes ou para outros tipos de destino de armazenamento ou processamento de mensagens. Os aliases são úteis, por exemplo, para colocar as mensagens enviadas para `postmaster@lab2.campus` na caixa de correio de Carol, que é uma caixa de correio local comum do sistema `lab2.campus`. Para isso, a linha `postmaster: carol` deve ser adicionada ao arquivo `/etc/aliases` em `lab2.campus`. Após modificar o arquivo `/etc/aliases`, o comando `newaliases` deve ser executado para atualizar os bancos de dados de aliases do MTA e efetivar as alterações. Os comandos `sendmail -bi` ou `sendmail -I` também servem para atualizar o banco de dados de aliases.

Os aliases são definidos em linhas, no formato `<alias>: <destination>`. Além das caixas de correio locais comuns, indicadas pelo nome de usuário correspondente, outros tipos de destino estão disponíveis:

- Um caminho completo (começando com `/`) para um arquivo. As mensagens enviadas para o alias correspondente serão anexadas ao arquivo.
- Um comando para processar a mensagem. `<destination>` deve começar com um caractere de pipe `|`, se o comando contiver caracteres especiais (como espaços em brancos), ele deve ser posto entre aspas duplas. Por exemplo, o alias `subscribe: | subscribe.sh` em `lab2.campus` encaminha todas as mensagens enviadas para `subscribe@lab2.campus` para a entrada padrão do comando `subscribe.sh`. Se `sendmail` estiver rodando no *modo restrito do shell*, os comandos permitidos — ou os links para eles — deverão estar em `/etc/smrsh/`.
- Um arquivo de inclusão. Um único alias pode ter diversos destinos (separados por vírgulas) e, portanto, pode ser mais prático mantê-los em um arquivo externo. A palavra-chave `:include:` deve indicar o caminho do arquivo, como em `:include:/var/local/destinations`
- Um endereço externo. Os aliases também podem encaminhar mensagens para endereços de email externos.
- Outro alias.

Um usuário local sem privilégios pode definir aliases para seu próprio email editando o arquivo `.forward` em seu diretório inicial. Como os aliases podem afetar apenas sua própria caixa de correio, somente a parte `<destination>` é necessária. Para encaminhar todos os emails recebidos para um endereço externo, por exemplo, o usuário `dave` em `lab2.campus` poderia criar o seguinte arquivo `~/.forward`:

```
$ cat ~/.forward
```

emma@lab1.campus

Ele encaminhará todas as mensagens de email enviadas para dave@lab2.campus para emma@lab1.campus. Como no caso do arquivo /etc/aliases, outras regras de redirecionamento podem ser adicionadas a .forward, uma por linha. Entretanto, o arquivo .forward deve ser gravável apenas por seu dono e não é necessário executar o comando newaliases após modificá-lo. Os arquivos que começam com um ponto não aparecem nas listagens de arquivos regulares, fazendo com que o usuário desconheça alguns dos aliases ativos. Portanto, é importante verificar se o arquivo existe ao diagnosticar problemas de entrega de email.

Exercícios Guiados

1. Sem outras opções ou argumentos, o comando `mail henry@lab3.campus` inicia o modo de inserção de dados para que o usuário possa digitar a mensagem para `henry@lab3.campus`. Depois de terminar a mensagem, qual pressionamento de tecla fecha o modo de inserção e envia o email?

2. Qual comando o usuário root pode executar para listar as mensagens não entregues que se originaram no sistema local?

3. Como um usuário sem privilégios pode usar o método MTA padrão para encaminhar automaticamente todos os seus emails recebidos para o endereço `dave@lab2.campus`?

Exercícios Exploratórios

1. Usando o comando `mail` fornecido por `mailx`, qual comando enviará uma mensagem para `emma@lab1.campus` com o arquivo `logs.tar.gz` como um anexo e a saída do comando `uname -a` como o corpo do email?

2. Um administrador de serviços de email deseja monitorar as transferências de email pela rede, mas não quer sobrecarregar sua caixa de correio com mensagens de teste. Como esse administrador poderia configurar um alias de email em todo o sistema de forma a redirecionar todos os emails enviados ao usuário `test` para o arquivo `/dev/null`?

3. Que comando, além de `newaliases`, poderia ser usado para atualizar o banco de dados de aliases após se adicionar um novo alias a `/etc/aliases`?

Resumo

Esta lição abordou a função e o uso de Mail Transfer Agents (Agentes de transferência de correio) nos sistemas Linux. O MTA fornece um método padrão para comunicação entre contas de usuário e pode ser combinado com outros softwares para fornecer funcionalidades extras. A lição discutiu os seguintes tópicos:

- Conceitos de tecnologias, caixas de correio e protocolos relacionados ao email.
- Como os MTAs do Linux trocam mensagens pela rede.
- Clientes de email e MUAs (Mail User Agents) no console.
- Aliases e encaminhamento de email local.

As tecnologias, comandos e procedimentos abordados foram:

- SMTP e protocolos relacionados.
- MTAs disponíveis para o Linux: Sendmail, Postfix, qmail, Exim.
- Comandos de MTA e MUA: `sendmail` e `mail`.
- Arquivos e comandos administrativos: `mailq`, `/etc/aliases`, `newaliases`, `~/.forward`.

Respostas aos Exercícios Guiados

1. Sem outras opções ou argumentos, o comando `mail henry@lab3.campus` inicia o modo de inserção de dados para que o usuário possa digitar a mensagem para `henry@lab3.campus`. Depois de terminar a mensagem, qual pressionamento de tecla fecha o modo de inserção e envia o email?

Pressionar `ctrl + D` fechará o programa e disparará o email.

2. Qual comando o usuário root pode executar para listar as mensagens não entregues que se originaram no sistema local?

O comando `mailq` ou `sendmail -bp`.

3. Como um usuário sem privilégios pode usar o método MTA padrão para encaminhar automaticamente todos os seus emails recebidos para o endereço `dave@lab2.campus`?

O usuário deve adicionar `dave@lab2.campus` em `~/forward`.

Respostas aos Exercícios Exploratórios

1. Usando o comando `mail` fornecido por `mailx`, qual comando enviará uma mensagem para `emma@lab1.campus` com o arquivo `logs.tar.gz` como um anexo e a saída do comando `uname -a` como o corpo do email?

```
uname -a | mail -a logs.tar.gz emma@lab1.campus
```

2. Um administrador de serviços de email deseja monitorar as transferências de email pela rede, mas não quer sobrecarregar sua caixa de correio com mensagens de teste. Como esse administrador poderia configurar um alias de email em todo o sistema de forma a redirecionar todos os emails enviados ao usuário `test` para o arquivo `/dev/null`?

A linha `test: /dev/null` em `/etc/aliases` redireciona todas as mensagens enviadas para a caixa de correio local `test` para o arquivo `/dev/null`.

3. Que comando, além de `newaliases`, poderia ser usado para atualizar o banco de dados de aliases após se adicionar um novo alias a `/etc/aliases`?

O comando `sendmail -bi` ou `sendmail -I`.



108.4 Configurar impressoras e impressão

Referência ao LPI objectivo

[LPIC-1 version 5.0, Exam 102, Objective 108.4](#)

Peso

2

Áreas chave de conhecimento

- Configuração básica do CUPS (para impressoras locais e remotas).
- Gerenciar a fila de impressão do usuário.
- Resolução de problemas gerais de impressão.
- Adicionar e remover trabalhos da fila de impressão de impressoras configuradas.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- Arquivos de configuração do CUPS, ferramentas e utilitários
- `/etc/cups/`
- Interface legada lpd (`lpr`, `lprm`, `lpq`)



**Linux
Professional
Institute**

108.4 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	108 Serviços essenciais do sistema
Objetivo:	108.4 Gerenciamento de impressoras e impressão
Lição:	1 de 1

Introdução

As afirmações sobre a chegada de uma “sociedade sem papel” após o advento dos computadores não se comprovaram até hoje. Muitas organizações ainda dependem de páginas de informações impressas, ou “no papel”. Assim, é obviamente importante para um usuário saber como imprimir a partir de um sistema, assim como um administrador precisa saber como manter a capacidade de um computador de trabalhar com impressoras.

No Linux, assim como em muitos outros sistemas operacionais, a pilha de software *Common Unix Printing System* (CUPS) permite a impressão e o gerenciamento de impressoras a partir de um computador. Aqui está um esboço muito simplificado de como um arquivo é impresso no Linux usando CUPS:

1. Um usuário envia um arquivo para impressão.
2. O daemon do CUPS, `cupsd`, então põe no *spool* (fila de espera) o trabalho de impressão. Ele recebe um número de trabalho do CUPS, junto com informações sobre a fila de impressão que contém o trabalho, bem como o nome do documento a ser impresso.

3. O CUPS utiliza *filtros* que são instalados no sistema para gerar um arquivo formatado que a impressora pode usar.
4. O CUPS então envia o arquivo reformatado à impressora para impressão.

Veremos essas etapas com mais detalhes, bem como a maneira de instalar e gerenciar uma impressora no Linux.

O serviço CUPS

A maioria das instalações do Linux em máquinas pessoais já vem com os pacotes do CUPS instalados. Em instalações mínimas do Linux, os pacotes do CUPS podem não estar presentes, dependendo da distribuição. Uma instalação básica do CUPS pode ser realizada em um sistema Debian com o seguinte:

```
$ sudo apt install cups
```

Em sistemas Fedora, o processo de instalação é igualmente fácil. É necessário iniciar o serviço CUPS manualmente após a instalação no Fedora e outras distribuições baseadas no Red Hat:

```
$ sudo dnf install cups
...
$ sudo systemctl start cups.service
```

Após a conclusão da instalação, você pode verificar se o serviço CUPS está sendo executado com o uso do comando `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://
```

Como acontece com muitos outros daemons do Linux, o CUPS depende de um conjunto de

arquivos de configuração para suas operações. Na lista abaixo estão os que são de maior interesse para o administrador do sistema:

/etc/cups/cupsd.conf

Este arquivo contém as configurações do próprio serviço CUPS. Se você estiver familiarizado com o arquivo de configuração do servidor web Apache, o arquivo de configuração do CUPS não lhe será estranho, pois ele usa uma sintaxe muito semelhante. O arquivo `cupsd.conf` contém configurações para coisas como controlar o acesso às diversas filas de impressão em uso no sistema, se a interface web do CUPS está habilitada ou não, bem como o nível de registro no log usado pelo daemon.

/etc/printcap

Este é o arquivo legado que foi usado pelo protocolo LPD (*Line Printer Daemon*) antes do advento do CUPS. O CUPS continua a criar este arquivo para garantir a compatibilidade com versões anteriores e muitas vezes ele é um link simbólico para `/run/cups/printcap`. Cada linha deste arquivo contém uma impressora à qual o sistema tem acesso.

/etc/cups/printers.conf

Este arquivo contém as impressoras configuradas para serem usadas pelo sistema CUPS. Neste arquivo, cada impressora e sua fila de impressão associada estão contidas em uma estrofe `<Printer></Printer>`. Este arquivo fornece as listagens individuais de impressoras encontradas em `/etc/printcap`.

WARNING

Nenhuma modificação deve ser feita no arquivo `/etc/cups/printers.conf` na linha de comando enquanto o serviço CUPS estiver em execução.

/etc/cups/ppd/

Este não é um arquivo de configuração, mas um diretório que contém os arquivos *PostScript Printer Description* (PPD) para as impressoras que os utilizam. A capacidade operacional de cada impressora será armazenada em um arquivo PPD (terminando com a extensão `.ppd`). Esses são arquivos de texto simples e seguem um formato específico.

O serviço CUPS também utiliza o log da mesma maneira que o serviço Apache 2. Os logs são armazenados em `/var/log/cups/` e contêm um `access_log`, um `page_log` e um `error_log`. O `access_log` mantém um registro de acesso à interface web do CUPS, bem como as ações realizadas dentro dela, como o gerenciamento da impressora. O `page_log` rastreia os trabalhos de impressão que foram submetidos às filas de impressão gerenciadas pela instalação do CUPS. O `error_log` contém as mensagens sobre trabalhos de impressão que falharam e outros erros registrados pela interface web.

A seguir, veremos as ferramentas e utilitários usados para gerenciar o serviço do CUPS.

Usando a interface web

Como já dissemos, o arquivo de configuração `/etc/cups/cupsd.conf` determina se a interface web do sistema CUPS está habilitada. A configuração é semelhante a esta:

```
# Web interface setting...
WebInterface Yes
```

Se a interface web estiver habilitada, o CUPS pode ser gerenciado a partir de um navegador na URL padrão `http://localhost:631`. Por padrão, um usuário do sistema pode visualizar as impressoras e filas de impressão, mas qualquer modificação da configuração requer que um usuário com acesso root se autentique no serviço web. A estrofe de configuração no arquivo `/etc/cups/cupsd.conf` para restringir o acesso aos recursos administrativos será semelhante à seguinte:

```
# 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>
```

Vamos olhar opções mais de perto:

AuthType Default

usa um prompt de autenticação básico quando uma ação requer acesso de root.

Require user @SYSTEM

indica que é necessário um usuário com privilégios administrativos para a operação. Pode ser alterado para `@groupname`, onde membros de `groupname` podem administrar o serviço do CUPS; ou ainda, usuários individuais podem ser listados, como em `Require user carol, tim`.

Order deny,allow

funciona de forma semelhante à opção de configuração do Apache 2, onde a ação é negada por padrão a menos que um usuário (ou membro de um grupo) seja autenticado.

A interface web do CUPS pode ser desabilitada primeiro interrompendo o serviço do CUPS, alterando a opção WebInterface de Yes para No e, finalmente, reiniciando o serviço do CUPS.

A interface web do CUPS é construída como um site básico, com guias de navegação para as diversas seções do sistema CUPS. Estas são as guias incluídas na interface:

Home

A página inicial lista a versão atual do CUPS que está instalada. Ela também divide o CUPS em seções como:

CUPS for Users

Fornece uma descrição do CUPS, opções de linha de comando para trabalhar com impressoras e filas de impressão e um link para o fórum de usuários do CUPS.

CUPS for Administrators

Fornece links na interface para instalar e gerenciar impressoras e links para informações sobre como trabalhar com impressoras em uma rede.

CUPS for Developers

Fornece links para o desenvolvimento do próprio CUPS, bem como para a criação de arquivos PPD para impressoras.

Administration

A página de administração também é dividida em seções:

Printers

Aqui, um administrador pode adicionar novas impressoras ao sistema, localizar as impressoras conectadas ao sistema e gerenciar as impressoras que já estão instaladas.

Classes

As classes são um mecanismo no qual impressoras podem ser adicionadas a grupos com políticas específicas. Por exemplo, uma classe pode conter um grupo de impressoras pertencentes a um andar específico de um prédio, na qual apenas os usuários de um determinado departamento podem imprimir. Outra classe pode ter limitações sobre quantas páginas um usuário pode imprimir. As classes não são criadas por padrão na instalação do CUPS e devem ser definidas por um administrador. Esta é a seção da interface web do CUPS na qual novas classes podem ser criadas e gerenciadas.

Jobs

Aqui é onde um administrador pode visualizar todos os trabalhos de impressão que estão atualmente na fila para todas as impressoras gerenciadas por esta instalação do CUPS.

Server

Aqui é onde um administrador pode fazer alterações no arquivo `/etc/cups/cupsd.conf`. Além disso, outras opções de configuração estão disponíveis por meio de caixas de seleção, como permitir que impressoras conectadas a esta instalação do CUPS sejam compartilhadas em uma rede, autenticação avançada e permitir a administração remota de impressoras.

Classes

Se houver classes de impressoras configuradas no sistema, elas serão listadas nesta página. Cada classe de impressora tem opções para gerenciar todas as impressoras da classe de uma vez, bem como visualizar todos os trabalhos que estão na fila para as impressoras daquela classe.

Help

Esta guia fornece links para toda a documentação disponível do CUPS instalado no sistema.

Jobs

A guia Jobs permite a pesquisa por trabalhos de impressão individuais, bem como a listagem de todos os trabalhos de impressão atuais gerenciados pelo servidor.

Printers

A guia Printers lista todas as impressoras atualmente gerenciadas pelo sistema, além de dar uma visão geral rápida do status de cada impressora. É possível clicar em cada impressora listada para ir à página na qual aquela impressora pode ser gerenciada mais detalhadamente. As informações sobre as impressoras nesta guia vêm do arquivo `/etc/cups/printers.conf`.

Instalando uma impressora

Adicionar uma impressora ao sistema é um processo simples na interface web do CUPS:

1. Clique na guia **Administration** e depois no botão **Add Printer**.
2. A página seguinte tem uma série de opções dependendo de como a sua impressora estiver conectada ao sistema. Se for uma impressora local, selecione a opção mais relevante, como a porta à qual a impressora está conectada ou o software de impressão de terceiros que estiver instalado. O CUPS também tenta detectar impressoras conectadas à rede e as exibe aqui. Também é possível escolher uma opção de conexão direta a uma impressora de rede, dependendo dos protocolos de impressão em rede suportados pela impressora. Selecione a opção apropriada e clique no botão **Continue**.
3. A página seguinte permite atribuir um nome, descrição e local (como “escritório” ou “recepção”) para a impressora. Se quiser compartilhar esta impressora pela rede, marque a

caixa de seleção dessa opção nesta página. Depois de inserir as configurações, clique no botão **Continue**.

4. A página seguinte é onde a marca e o modelo da impressora podem ser selecionados. Essa informação permite que o CUPS pesquise em seu banco de dados local pelos drivers e arquivos PPD mais adequados para usar com a impressora. Se você tiver um arquivo PPD fornecido pelo fabricante da impressora, navegue até o local em que ele está e selecione-o. Depois disso, clique no botão **Add Printer**.
5. A última página é onde são definidas as opções padrão, como o tamanho da página usada pela impressora e a resolução dos caracteres impressos. Clique no botão **Set Default Options** e a impressora estará instalada no sistema.

NOTE

Muitas instalações pessoais do Linux têm diferentes ferramentas que podem ser usadas para instalar uma impressora. Os ambientes de desktop GNOME e KDE possuem seus próprios aplicativos integrados para instalar e gerenciar impressoras. Além disso, algumas distribuições fornecem aplicativos separados de gerenciamento de impressoras. No entanto, ao trabalhar com uma instalação de servidor que muitos usuários acessarão para seus trabalhos de impressão, a interface web do CUPS fornece as melhores ferramentas para a tarefa.

Uma fila de impressão também pode ser instalada usando os comandos LPD/LPR legados. Eis um exemplo usando o comando `lpadmin`:

```
$ sudo lpadmin -p ENVY-4510 -L "office" -v socket://192.168.150.25 -m everywhere
```

Vamos detalhar o comando para ilustrar as opções usadas aqui:

- Como a adição de uma impressora ao sistema requer um usuário com privilégios administrativos, acrescentamos o comando `lpadmin` com `sudo`.
- A opção `-p` é o destino de seus trabalhos de impressão. É, essencialmente, um nome amigável para o usuário saber para onde vão os trabalhos de impressão. Normalmente, você pode fornecer o nome da impressora.
- A opção `-L` é a localização da impressora. Esse dado é opcional, mas útil quando se gerenciam várias impressoras em diferentes locais.
- A opção `-v` é para o URI do dispositivo de impressão. A fila de impressão do CUPS precisa do URI do dispositivo para enviar os trabalhos de impressão já preparados para uma impressora específica. Em nosso exemplo, estamos usando um local de rede empregando o endereço IP fornecido.
- A opção final, `-m`, está definida como “everywhere” (em todos os lugares). Essa opção define o

modelo da impressora para o CUPS poder determinar qual arquivo PPD usar. Nas versões modernas do CUPS, é melhor usar “everywhere” para que o CUPS verifique o URI do dispositivo (definido na opção `-v` anterior) e determine assim automaticamente o arquivo PPD correto a ser usado para a impressora. Em situações modernas, o CUPS usará apenas o IPP, conforme explicado abaixo.

Como já dissemos, é melhor deixar o CUPS determinar automaticamente qual arquivo PPD usar para uma determinada fila de impressão. No entanto, o comando `lpinfo` legado pode ser usado para consultar os arquivos PPD instalados localmente para ver o que está disponível. Basta fornecer a opção `--make-and-model` (marca e modelo) da impressora que deseja instalar, mais a opção `-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
```

Note que o comando `lpinfo` está obsoleto. Ele foi mostrado aqui como um exemplo de listagem dos arquivos de driver de impressão que uma impressora pode usar.

WARNING

As versões futuras do CUPS têm drivers obsoletos e, por isso, se concentrarão no uso de IPP (*Internet Printing Protocol*) e formatos de arquivo padrão. A saída do comando anterior ilustra isso com a capacidade de impressão `everywhere IPP Everywhere`. O IPP pode executar as mesmas tarefas de um driver de impressão. O IPP, assim como a interface web do CUPS, utiliza a porta de rede 631 com o protocolo TCP.

Uma impressora padrão pode ser definida usando o comando `lpoptions`. Desta forma, se a maioria (ou todos) os trabalhos de impressão forem enviados para uma impressora em particular, aquela que foi especificada com o comando `lpoptions` será a padrão. Basta especificar a impressora junto com a opção `-d`:

```
$ lpoptions -d ENVY-4510
```

Gerenciando impressoras

Depois que a impressora é instalada, o administrador pode usar a interface web para gerenciar as opções disponíveis para a impressora. Uma abordagem mais direta para gerenciar uma

impressora é o comando `lpadmin`.

Uma opção é permitir que uma impressora seja compartilhada na rede. Isso pode ser feito com a opção `printer-is-shared`, especificando-se a impressora com a opção `-p`:

```
$ sudo lpadmin -p FRONT-DESK -o printer-is-shared=true
```

O administrador também pode configurar uma fila de impressão para aceitar apenas trabalhos de impressão de usuários específicos, com cada usuário separado por uma vírgula:

```
$ sudo lpadmin -p FRONT-DESK -u allow:carol,frank,grace
```

Inversamente, podemos negar o acesso a uma fila de impressão específica a determinados usuários:

```
$ sudo lpadmin -p FRONT-DESK -u deny:dave
```

Também é possível permitir ou negar acesso à fila de impressão a grupos de usuários, desde que o nome do grupo seja precedido pelo caractere “arroba” (@):

```
$ sudo lpadmin -p FRONT-DESK -u deny:@sales,@marketing
```

Uma fila de impressão também pode ter uma política de erro caso encontre problemas ao imprimir um trabalho. Com o uso de políticas, um trabalho de impressão pode ser abortado (`abort-job`) ou outra tentativa de impressão pode ocorrer posteriormente (`retry-job`). Outras políticas incluem a capacidade de parar a impressora imediatamente caso ocorra um erro (`stop-printer`), bem como a possibilidade de repetir o trabalho imediatamente após a detecção de uma falha (`retry-current-job`). Eis um exemplo no qual a política da impressora foi definida para abortar o trabalho de impressão caso ocorra um erro na impressora `FRONT-DESK`:

```
$ sudo lpadmin -p FRONT-DESK -o printer-error-policy=abort-job
```

Consulte as páginas de manual do comando `lpadmin` localizadas em `lpadmin(8)` para obter mais detalhes sobre o uso deste comando.

Enviando trabalhos de impressão

Muitos aplicativos para desktop permitem enviar trabalhos de impressão a partir de um item de menu ou usando o atalho de teclado `ctrl + p`. Se você estiver em um sistema Linux que não utiliza um ambiente de desktop, ainda pode enviar arquivos para uma impressora por meio dos comandos LPD/LPR legados.

O comando `lpr` (“line printer remote”) é usado para enviar um trabalho de impressão para a fila da impressora. Em sua forma mais básica, basta incluir um nome de arquivo junto com o comando `lpr`:

```
$ lpr report.txt
```

O comando acima envia o arquivo `report.txt` para a fila de impressão padrão do sistema (identificada pelo arquivo `/etc/cups/printers.conf`).

Se uma instalação do CUPS tiver várias impressoras instaladas, o comando `lpstat` pode ser usado para imprimir uma lista de impressoras disponíveis usando a opção `-p`. A opção `-d` indica qual é a impressora padrão:

```
$ 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
```

Portanto, em nosso exemplo, o arquivo `report.txt` será enviado para a impressora `ENVY-4510`, pois ela está definida como padrão. Caso o arquivo precise ser impresso em uma impressora diferente, especifique a impressora junto com a opção `-P`:

```
$ lpr -P FRONT-DESK report.txt
```

Quando um trabalho de impressão é enviado ao CUPS, o daemon decide qual backend é mais adequado para a tarefa. O CUPS pode usar diversos drivers de impressora, filtros, monitores de porta de hardware e outros softwares para preparar o documento adequadamente. Haverá momentos em que o usuário que quiser imprimir um documento precisará fazer modificações na maneira *como* o documento será impresso. Certos aplicativos gráficos facilitam essa tarefa. Existem também opções em linha de comando que servem para alterar a forma como um

documento deve ser impresso. Quando um trabalho de impressão é enviado através da linha de comando, a opção `-o` (de “options”) pode ser usada em conjunto com termos específicos para ajustar o layout do documento para impressão. Eis uma pequena lista das opções comumente usadas:

landscape

O documento é impresso em modo paisagem, ou seja, com a página rotacionada 90 graus no sentido horário. A opção `orientation-requested=4` dá o mesmo resultado.

two-sided-long-edge

A impressora imprime o documento em modo retrato em ambos os lados do papel, desde que a impressora suporte este recurso

two-sided-short-edge

A impressora imprime o documento em modo paisagem em ambos os lados do papel, desde que a impressora suporte este recurso

media

A impressora imprime o trabalho no tamanho de mídia especificado. Os tamanhos de mídia disponíveis para um trabalho de impressão dependem da impressora, mas estes são os tamanhos mais comuns:

Size Option	Purpose
A4	ISO A4
Letter	US Letter
Legal	US Legal
DL	ISO DL Envelope
COM10	US #10 Envelope

collate

Agrupar o documento impresso. Isso é útil quando precisamos imprimir um documento de várias páginas mais de uma vez, pois todas as páginas de cada documento serão impressas na ordem. Defina esta opção como `true` para ativá-la ou ` `false` ` para desativá-la.

page-ranges

Esta opção pode ser usada para selecionar uma única página ou um conjunto específico de páginas a imprimir. Um exemplo seria: `-o page-ranges=5-7,9,15`. Seriam impressas as páginas 5, 6 e 7 e as páginas 9 e 15.

fit-to-page

Imprime o documento redimensionando o arquivo de forma a caber no papel. Se nenhuma informação sobre o tamanho da página for fornecida pelo arquivo a ser impresso, é possível que o trabalho seja dimensionado incorretamente e partes do documento podem acabar ficando para fora da página, ou o documento ficar pequeno demais.

outputorder

Imprime o documento na ordem `reverse` (inversa) ou `normal`, ou seja, iniciando a impressão na página um. Se uma impressora imprimir as páginas com o rosto para baixo, o padrão é que a ordem seja `-o outputorder=normal`, ao passo que as impressoras que imprimem com as páginas voltadas para cima imprimirão com `-o outputorder=reverse`.

Tomando uma amostra das opções acima, o seguinte comando de exemplo pode ser construído:

```
$ lpr -P ACCOUNTING-LASERJET -o landscape -o media=A4 -o two-sided-short-edge finance-report.pdf
```

Para imprimir mais de uma cópia de um documento, usamos a opção de número no seguinte formato: `-#N`, onde `N` é igual ao número de cópias a serem impressas. Eis um exemplo com a opção de agrupamento, em que sete cópias de um relatório devem ser impressas na impressora padrão:

```
$ lpr -#7 -o collate=true status-report.pdf
```

Além do comando `lpr`, o comando `lp` também pode ser usado. Muitas das opções usadas com `lpr` também funcionam com o comando `lp`, mas há algumas diferenças. Consulte a página de manual em `lp(1)` para referência. Veja como podemos executar o exemplo anterior do comando `lpr` usando a sintaxe do comando `lp`, especificando também a impressora de destino com a opção `-d`:

```
$ lp -d ACCOUNTING-LASERJET -n 7 -o collate=true status-report.pdf
```

Gerenciando trabalhos de impressão

Como já dissemos, cada trabalho enviado à fila de impressão recebe uma ID de trabalho do CUPS. Um usuário pode visualizar os trabalhos de impressão que enviou com o comando `lpq`. A opção `-a` mostra as filas de todas as impressoras gerenciadas pela instalação do CUPS:

\$ lpq -a				
Rank	Owner	Job	File(s)	Total Size

1st	carol	20	finance-report.pdf	5072 bytes
-----	-------	----	--------------------	------------

O mesmo comando `lpstat` usado anteriormente também inclui uma opção para visualizar as filas de impressão. A opção `-o` sozinha mostra todas as filas de impressão; uma fila de impressão também pode ser especificada pelo nome:

```
$ lp -o
ACCOUNTING-LASERJET-4 carol 19456 Wed 05 Aug 2020 04:29:44 PM EDT
```

A ID do trabalho de impressão será anexada ao nome da fila para onde o trabalho foi enviado, seguida pelo nome do usuário que enviou o trabalho, o tamanho do arquivo e a hora em que foi enviado.

Se um trabalho de impressão travar em uma impressora ou se um usuário desejar cancelar seu trabalho de impressão, usamos o comando `lprm` junto com o ID do trabalho encontrado no comando `lpq`:

```
$ lprm 20
```

Todos os trabalhos de uma fila de impressão podem ser excluídos de uma vez com apenas um hífen `-`:

```
$ lprm -
```

Alternativamente, o comando `cancel` do CUPS também pode ser empregado por um usuário para interromper seu trabalho de impressão atual:

```
$ cancel
```

Um trabalho de impressão específico pode ser cancelado por sua ID de trabalho precedida pelo nome da impressora:

```
$ cancel ACCOUNTING-LASERJET-20
```

Um trabalho de impressão também pode ser movido de uma fila de impressão para outra. Isso geralmente é útil se uma impressora parar de responder ou se o documento a ser impresso exigir recursos disponíveis em uma impressora diferente. Observe que esse procedimento normalmente

requer um usuário com privilégios elevados. Usando o mesmo trabalho de impressão do exemplo anterior, podemos movê-lo para a fila da impressora FRONT-DESK:

```
$ sudo lpmove ACCOUNTING-LASERJET-20 FRONT-DESK
```

Removendo impressoras

Para remover uma impressora, é interessante listar primeiro todas as impressoras gerenciadas atualmente pelo serviço do CUPS. Usamos para isso o comando 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
```

A opção **-v** não apenas lista as impressoras, mas também onde (e como) elas estão conectadas. É aconselhável rejeitar primeiro todos os novos trabalhos que vão para essa impressora e incluir um motivo para a impressora não aceitar novos trabalhos. Fazemos isso desta forma:

```
$ sudo cupsreject -r "Printer to be removed" FRONT-DESK
```

Note o uso de **sudo**, já que esta tarefa requer um usuário com privilégios elevados.

Para remover uma impressora, utilizamos o comando lpadmin com a opção **-x**:

```
$ sudo lpadmin -x FRONT-DESK
```

Exercícios Guiados

1. Uma nova impressora acabou de ser instalada em uma estação de trabalho local chamada `office-mgr`. Que comando pode ser usado para definir esta impressora como padrão para esta estação de trabalho?

2. Qual comando e opção usariamos para determinar quais impressoras estão disponíveis para impressão em uma estação de trabalho?

3. Usando o comando `cancel`, como você removeria um trabalho de impressão com ID 15 que está preso na fila da impressora chamada `office-mgr`?

4. Temos um trabalho de impressão destinado a uma impressora que não possui papel suficiente para imprimir o arquivo completo. Que comando você usaria para mover o trabalho de impressão com ID 2 da fila de impressão da impressora `FRONT-DESK` para a fila de impressão da impressora `ACCOUNTING-LASERJET`?

Exercícios Exploratórios

Usando o gerenciador de pacotes de sua distribuição, instale os pacotes `cups` e `printer-driver-cups-pdf`. Note que, se você estiver usando uma distribuição baseada em Red Hat (como o Fedora), o driver CUPS PDF é chamado de `cups-pdf`. Instale também o pacote `cups-client` para utilizar os comandos de impressão no estilo System V. Usaremos esses pacotes para praticar o gerenciamento de uma impressora CUPS sem instalar fisicamente uma impressora real.

1. Verifique se o daemon do CUPS está em execução e, em seguida, confira se a impressora PDF está ativada e definida como padrão.

2. Execute um comando para imprimir o arquivo `/etc/services`. Você deverá obter um diretório chamado PDF dentro do seu diretório inicial.

3. Use um comando que desabilite somente a impressora e execute um comando separado que mostre todas as informações de status para verificar se a impressora PDF está desabilitada. Em seguida, tente imprimir uma cópia de seu arquivo `/etc/fstab`. O que acontece?

4. Agora, tente imprimir uma cópia do arquivo `/etc/fstab` na impressora PDF. O que acontece?

5. Cancele o trabalho de impressão e remova a impressora PDF.

Resumo

O daemon do CUPS é uma plataforma amplamente usada para impressão em impressoras locais e remotas. Embora tenha substituído o protocolo LPD legado, ele ainda fornece compatibilidade reversa para suas ferramentas.

Os arquivos e comandos discutidos nesta lição foram:

/etc/cups/cupsd.conf

O arquivo de configuração principal para o próprio serviço do CUPS. Este arquivo também controla o acesso à interface web do CUPS.

/etc/printcap

Um arquivo legado usado pelo LPD contendo uma linha para cada impressora conectada ao sistema.

/etc/cups/printers.conf

O arquivo de configuração usado pelo CUPS para informações sobre as impressoras.

A interface web do CUPS, que em uma instalação padrão pode ser encontrada em <http://localhost:631>. Lembre-se de que a porta de rede padrão para a interface web é 631/TCP.

Os seguintes comandos LPD/LPR legados também foram discutidos:

lpadmin

Usado para instalar e remover impressoras e classes de impressoras

lpoptions

Usado para exibir as opções da impressora e modificar as configurações de uma impressora.

lpstat

Usado para exibir informações de status das impressoras conectadas a uma instalação do CUPS.

lpr

Usado para enviar trabalhos de impressão para uma fila.

lp

Usado para enviar trabalhos de impressão para uma fila.

lpq

lista os trabalhos de impressão na fila.

lprm

Usado para cancelar trabalhos de impressão por ID. O ID de um trabalho pode ser obtido com a saída do comando `lpq`.

cancel

Uma alternativa ao comando `lprm` para cancelar trabalhos de impressão pelo ID.

Não deixe de estudar as seguintes páginas de manual para as diversas ferramentas e utilitários do CUPS: `lpadmin(8)`, `lpoptions(1)`, `lpr(1)`, `lpq(1)`, `lprm(1)`, `cancel(1)`, `lpstat(1)`, `cupsenable(8)` e `cupsaccept(8)`. Consulte também a documentação de ajuda online em <http://localhost:631/help>.

Respostas aos Exercícios Guiados

- Uma nova impressora acabou de ser instalada em uma estação de trabalho local chamada `office-mgr`. Que comando pode ser usado para definir esta impressora como padrão para esta estação de trabalho?

```
$ lpoptions -d office-mgr
```

- Qual comando e opção usariam para determinar quais impressoras estão disponíveis para impressão em uma estação de trabalho?

```
$ lpstat -p
```

A opção `-p` lista todas as impressoras disponíveis e informa se elas estão habilitadas para impressão.

- Usando o comando `cancel`, como você removeria um trabalho de impressão com ID 15 que está preso na fila da impressora chamada `office-mgr`?

```
$ cancel office-mgr-15
```

- Temos um trabalho de impressão destinado a uma impressora que não possui papel suficiente para imprimir o arquivo completo. Que comando você usaria para mover o trabalho de impressão com ID 2 da fila de impressão da impressora `FRONT-DESK` para a fila de impressão da impressora `ACCOUNTING-LASERJET`?

```
$ sudo lpmove FRONT-DESK-2 ACCOUNTING-LASERJET
```

Respostas aos Exercícios Exploratórios

Usando o gerenciador de pacotes de sua distribuição, instale os pacotes `cups` e `printer-driver-cups-pdf`. Note que, se você estiver usando uma distribuição baseada em Red Hat (como o Fedora), o driver CUPS PDF é chamado de `cups-pdf`. Instale também o pacote `cups-client` para utilizar os comandos de impressão no estilo System V. Usaremos esses pacotes para praticar o gerenciamento de uma impressora CUPS sem instalar fisicamente uma impressora real.

1. Verifique se o daemon do CUPS está em execução e, em seguida, confira se a impressora PDF está ativada e definida como padrão.

Um método para verificar a disponibilidade e o status da impressora PDF seria executar o seguinte comando:

```
$ lpstat -p -d
printer PDF is idle. enabled since Thu 25 Jun 2020 02:36:07 PM EDT
system default destination: PDF
```

2. Execute um comando para imprimir o arquivo `/etc/services`. Você deverá obter um diretório chamado PDF dentro do seu diretório inicial.

```
$ lp -d PDF /etc/services
```

funcionaria. Nesse momento você terá uma versão em PDF deste arquivo no diretório PDF.

3. Use um comando que desabilite somente a impressora e execute um comando separado que mostre todas as informações de status para verificar se a impressora PDF está desabilitada.

```
$ sudo cupsdisable PDF
```

desabilita a impressora.

Em seguida, execute o comando `lpstat -t` para obter uma lista completa das condições da impressora. Deve ser semelhante à saída a seguir:

```
$ 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. Agora, tente imprimir uma cópia do arquivo `/etc/fstab` na impressora PDF. O que acontece?

Depois de rodar o comando `lp -d PDF /etc/fstab`, obtemos uma saída mostrando as informações de ID do trabalho. No entanto, se você verificar a pasta PDF em seu diretório inicial, o novo arquivo não estará lá. Mas se verificarmos a fila de impressão com o comando `lpstat -o`, o trabalho estará listado ali.

5. Cancele o trabalho de impressão e remova a impressora PDF.

Usando a saída do comando `lp` anterior, use o comando `cancel` para excluir o trabalho. Por exemplo:

```
$ cancel PDF-4
```

Em seguida, execute o comando `lpstat -o` para verificar se o trabalho foi excluído.

Remova a impressora PDF com o seguinte comando: `sudo lpadmin -x PDF`. Por fim, verifique se a impressora foi removida: `lpstat -a`.



Tópico 109: Fundamentos de Rede



Linux
Professional
Institute

109.1 Fundamentos de protocolos de internet

Referência ao LPI objectivo

[LPIC-1 version 5.0, Exam 102, Objective 109.1](#)

Peso

4

Áreas chave de conhecimento

- Demonstrar um conhecimento adequado sobre máscaras de rede e a notação CIDR.
- Conhecimento sobre as diferenças entre endereços públicos de IP e reservados para uso de redes privadas (notação “dotted quad”).
- Conhecimento sobre as portas e serviços TCP e UDP mais comuns (20, 21, 22, 23, 25, 53, 80, 110, 123, 139, 143, 161, 162, 389, 443, 465, 514, 636, 993, 995).
- Conhecimento sobre as diferenças e principais características dos protocolos UDP, TCP e ICMP.
- Conhecimento das principais diferenças entre IPv4 e IPv6.
- Conhecimento sobre as características básicas do IPv6.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- /etc/services
- IPv4, IPv6
- Subredes
- TCP, UDP, ICMP



109.1 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	109 Fundamentos de rede
Objetivo:	109.1 Fundamentos dos protocolos de internet
Lição:	1 de 2

Introdução

O TCP/IP (*Transmission Control Protocol/Internet Protocol*, ou Protocolo de controle de transmissão/Protocolo de internet) é uma pilha de protocolos usados para permitir a comunicação entre computadores. Apesar do nome, a pilha é composta por diversos protocolos, como IP, TCP, UDP, ICMP, DNS, SMTP, ARP e outros.

IP (Internet Protocol)

O IP é o protocolo responsável pelo endereçamento lógico de um host, permitindo que o pacote seja enviado de um host para outro. Para isso, cada dispositivo da rede recebe um endereço IP exclusivo. É possível atribuir mais de um endereço ao mesmo dispositivo.

Na versão 4 do protocolo IP, normalmente denominado IPv4, o endereço é formado por um conjunto de 32 bits separados em 4 grupos de 8 bits, representados na forma decimal, denominados “dotted quad” (quadra pontilhada). Por exemplo:

Formato binário (4 grupos de 8 bits)

11000000.10101000.00001010.00010100

Formato decimal

192.168.10.20

No IPv4, os valores de cada octeto podem variar de 0 a 255, que é o equivalente a 11111111 em formato binário.

Classes de endereço

Teoricamente, os endereços IP são separados em classes, definidas pelo intervalo do primeiro octeto, conforme mostrado na tabela abaixo:

Classe	Primeiro octeto	Intervalo	Exemplo
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

IPs públicos e privados

Como já dissemos, para que a comunicação ocorra, cada dispositivo da rede deve estar associado a pelo menos um endereço IP exclusivo. Porém, se cada dispositivo conectado à Internet no mundo tivesse um endereço IP exclusivo, não haveria IPs suficientes (v4) para todos. Por causa disso, foram definidos endereços IP *privados*.

IPs privados são intervalos de endereços IP reservados para uso em redes internas (privadas) de empresas, instituições, residências, etc. Dentro da mesma rede, o uso de um endereço IP permanece exclusivo. No entanto, o mesmo endereço IP privado pode ser usado em qualquer rede privada.

Assim, na internet o tráfego de dados emprega endereços IP públicos, que são reconhecíveis e roteados pela internet, ao passo que nas redes privadas são usados esses intervalos de IP reservados. O roteador é responsável por converter o tráfego da rede privada para a rede pública e vice-versa.

Os intervalos de IPs privados, divididos em classes, podem ser vistos na tabela abaixo:

Classe	Primeiro octeto	Intervalo	IPs privados
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

Convertendo o formato decimal em binário

Para os temas deste tópico, é importante saber como converter endereços IP entre formatos binários e decimais.

A conversão do formato decimal para binário é feita por meio de divisões consecutivas por 2. Como exemplo, vamos converter o valor 105 empregando seguintes etapas:

1. Ao dividir o valor 105 por 2, temos:

```
105/2
Quociente = 52
Resto = 1
```

2. Dividimos o quociente sequencialmente por 2, até que o quociente seja igual a 1:

```
52/2
Resto = 0
Quociente = 26
```

```
26/2
Resto = 0
Quociente = 13
```

```
13/2
Resto = 1
Quociente = 6
```

```
6/2
```

Resto = 0
Quociente = 3

3/2
Resto = 1
Quociente = 1

- Agrupamos o último quociente seguido pelo resto de todas as divisões:

1101001

- Incluímos 0s à esquerda até chegar a 8 bits:

01101001

- No final, descobrimos que o valor 105 em decimal é igual a 01101001 em binário.

Convertendo o formato binário em decimal

Neste exemplo, usaremos o valor binário 10110000.

- Cada bit é associado a um valor com uma potência de base dois. As potências se iniciam em 0 e são incrementadas da direita para a esquerda. Neste exemplo teremos:

1	0	1	1	0	0	0	0
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

- Quando o bit é 1, atribuímos o valor da potência respectiva; quando o bit é 0, o resultado é 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

- Somamos todos os valores:

$$128 + 32 + 16 = 176$$

- Assim, 10110000 em binário é igual a 176 em decimal.

Netmask

A máscara de rede (ou *netmask*) é usada em conjunto com o endereço IP para determinar qual parte do IP representa a rede e qual representa os hosts. Ela tem o mesmo formato do endereço IP, ou seja, 32 bits em 4 grupos de 8. Por exemplo:

Decimal	Binário	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

Usando a máscara 255.255.0.0 como exemplo, podemos ver que, no IP associado a ela, os primeiros 16 bits (2 primeiros decimais) identificam a rede/sub-rede e os últimos 16 bits são usados para identificar exclusivamente os hosts dentro da rede.

O CIDR (*Classless Inter-Domain Routing*) mencionado acima está relacionado a uma notação de máscara simplificada, que indica o número de bits (1) associados à rede/sub-rede. Essa notação é comumente usada para substituir o formato decimal, por exemplo /24 em vez de 255.255.255.0.

Vale notar que cada classe de IP possui uma máscara padrão, da seguinte maneira:

Classe	Primeiro octeto	Intervalo	Máscara padrão
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

No entanto, esse padrão não significa que essa máscara sempre será usada. É possível usar qualquer máscara com qualquer endereço IP, como veremos a seguir.

Aqui estão alguns exemplos de uso de IPs e máscaras:

192.168.8.12 / 255.255.255.0 / 24

Intervalo

192.168.8.0 - 192.168.8.255

Endereço de rede

192.168.8.0

Endereço de transmissão

192.168.8.255

Hosts

192.168.8.1 - 192.168.8.254

Neste caso, temos os 3 primeiros dígitos (primeiros 24 bits) do endereço IP definindo a rede; o dígito final identifica os endereços dos hosts, ou seja, o intervalo desta rede vai de 192.168.8.0 a 192.168.8.255.

Chegamos assim a dois conceitos importantes: cada rede/sub-rede tem 2 endereços reservados. O primeiro endereço no intervalo é chamado de *endereço da rede*. Neste caso, 192.168.8.0, usado para identificar a própria rede/sub-rede. O último endereço no intervalo é chamado de *endereço de transmissão*, neste caso 192.168.8.255. Esse endereço de destino é usado para enviar a mesma mensagem (pacote) para todos os hosts IP dessa rede/sub-rede.

Os endereços de rede e de transmissão não podem ser usados pelas máquinas da rede. Portanto, a lista de IPs que podem de fato ser configurados varia de 192.168.8.1 a 192.168.8.254.

Vejamos um exemplo com mesmo IP, mas com uma máscara diferente:

192.168.8.12 / 255.255.0.0 / 16

Intervalo

192.168.0.0 - 192.168.255.255

Endereço de rede

192.168.0.0

Endereço de transmissão

192.168.255.255

Hosts

192.168.0.1 – 192.168.255.254

Perceba como a máscara diferente altera o intervalo de IPs que estão na mesma rede/sub-rede.

A divisão das redes por máscaras não se restringe aos valores padrão (8, 16, 24). Podemos criar subdivisões conforme desejado, adicionando ou removendo bits na identificação da rede, criando as novas sub-redes.

Por exemplo:

11111111.11111111.11111111.00000000 = 255.255.255.0 = 24

Se quisermos subdividir a rede acima em 2, basta adicionar outro bit à identificação da rede na máscara, desta maneira:

11111111.11111111.11111111.10000000 = 255.255.255.128 = 25

Teremos então as seguintes sub-redes:

192.168.8.0 - 192.168.8.127
192.168.8.128 - 192.168.8.255

Se aumentarmos ainda mais a subdivisão da rede:

11111111.11111111.11111111.11000000 = 255.255.255.192 = 26

Teremos:

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

Observe que em cada sub-rede teremos os endereços de rede reservados (a primeira do intervalo) e de transmissão (o último do intervalo); portanto, quanto mais subdividida a rede, menos IPs poderão ser usados efetivamente pelos hosts.

Identificando os endereços de rede e de transmissão

Por meio de um endereço IP e uma máscara, podemos identificar o endereço de rede e o endereço de transmissão, e assim definir a faixa de IPs para a rede/sub-rede.

O endereço de rede é obtido usando um “E lógico” entre o endereço IP e a máscara em seus formatos binários. Vejamos um exemplo usando o IP 192.168.8.12 e a máscara 255.255.255.192.

Convertendo do formato decimal para o binário, como vimos anteriormente, temos:

```
11000000.10101000.00001000.00001100 (192.168.8.12)
11111111.11111111.11111111.11000000 (255.255.255.192)
```

Com o “E lógico”, temos 1 e 1 = 1, 0 e 0 = 0, 1 e 0 = 0, então:

```
11000000.10101000.00001000.00001100 (192.168.8.12)
11111111.11111111.11111111.11000000 (255.255.255.192)
11000000.10101000.00001000.00000000
```

Portanto, o endereço de rede para essa sub-rede é 192.168.8.0.

Agora, para obter o endereço de transmissão, usamos o endereço de rede, forçando todos os bits relacionados ao host a ser 1s:

```
11000000.10101000.00001000.00000000 (192.168.8.0)
11111111.11111111.11111111.11000000 (255.255.255.192)
11000000.10101000.00001000.00111111
```

O endereço de transmissão, nesse caso, é 192.168.8.63.

Para concluir, temos:

```
192.168.8.12 / 255.255.255.192 / 26
```

Intervalo

```
192.168.8.0 - 192.168.8.63
```

Endereço de rede

192.168.8.0

Endereço de transmissão

192.168.8.63

Hosts

192.168.8.1 – 192.168.8.62

Rota padrão

Como vimos até agora, as máquinas que estão na mesma rede/sub-rede lógica podem se comunicar diretamente por meio do protocolo IP.

Mas vamos considerar o exemplo abaixo:

Rede 1

192.168.10.0/24

Rede 2

192.168.200.0/24

Neste caso, a máquina 192.168.10.20 não pode enviar diretamente um pacote para 192.168.200.100, pois elas estão em redes lógicas diferentes.

Para habilitar essa comunicação, usamos um roteador (ou um conjunto de roteadores). Um roteador nesta configuração também pode ser chamado de gateway (portal), pois fornece um portal entre duas redes. Esse dispositivo tem acesso a ambas as redes, pois está configurado com o IP das duas; por exemplo, 192.168.10.1 e 192.168.200.1. Por isso, ele é capaz de intermediar essa comunicação.

Para habilitar essa solução, cada host da rede deve ter configurado o que chamamos de *rota padrão*. A rota padrão indica o IP para o qual devem ser enviados todos os pacotes cujo destino é um IP que não faz parte da rede lógica do host.

No exemplo acima, a rota padrão para as máquinas na rede 192.168.10.0 / 24 será o IP 192.168.10.1, que é o IP do roteador/gateway, ao passo que a rota padrão para as máquinas na rede 192.168.200.0/24 será 192.168.200.1.

A rota padrão também é utilizada para que as máquinas da rede privada (LAN) tenham acesso à Internet (WAN) por meio de um roteador.

Exercícios Guiados

1. Usando o IP 172.16.30.230 e a máscara de rede 255.255.255.224, identifique:

A notação CIDR da máscara de rede	
Endereço de rede	
Endereço de transmissão	
Número de IPs que podem ser usados para os hosts nesta sub-rede	

2. Qual configuração é necessária em um host para permitir uma comunicação de IP com um host em uma rede lógica diferente?

Exercícios Exploratórios

1. Por que os intervalos de IP começando com 127 e o intervalo após 224 não estão incluídos nas classes de endereços IP A, B ou C?

2. Um dos campos mais importantes pertencentes a um pacote IP é o TTL (*Time To Live*). Qual é a função deste campo e como ele funciona?

3. Explique a função do NAT e quando ele é usado.

Resumo

Esta lição abordou os principais conceitos do protocolo IPv4, responsável por permitir a comunicação entre hosts em uma rede.

Também vimos as principais operações que o profissional deve conhecer para converter os IPs em diferentes formatos, além de aprender como analisar e realizar as configurações lógicas em redes e sub-redes.

Os seguintes assuntos foram abordados:

- Classes de endereços IP
- IPs públicos e privados
- Como converter IPs do formato decimal para o binário e vice-versa
- A máscara de rede (netmask)
- Como identificar os endereços de rede e de transmissão a partir do IP e máscara de rede
- Rota Padrão

Respostas aos Exercícios Guiados

1. Usando o IP 172.16.30.230 e a máscara de rede 255.255.255.224, identifique:

A notação CIDR da máscara de rede	27
Endereço de rede	172.16.30.224
Endereço de transmissão	172.16.30.255
Número de IPs que podem ser usados para os hosts nesta sub-rede	30

2. Qual configuração é necessária em um host para permitir uma comunicação de IP com um host em uma rede lógica diferente?

Rota Padrão

Respostas aos Exercícios Exploratórios

1. Por que os intervalos de IP começando com 127 e o intervalo após 224 não estão incluídos nas classes de endereços IP A, B ou C?

O intervalo que começa com 127 é reservado para endereços de loopback, usados para testes e comunicação interna entre processos, como o endereço 127.0.0.1. Além disso, os endereços acima de 224 também não são usados como endereços de host, mas para multicast e outros fins.

2. Um dos campos mais importantes pertencentes a um pacote IP é o TTL (*Time To Live*). Qual é a função deste campo e como ele funciona?

O TTL define a vida útil de um pacote. Ele é implementado por meio de um contador, no qual o valor inicial definido na origem é diminuído em cada gateway/roteador pelo qual o pacote passa, também chamado de “salto”. Se esse contador chegar a 0, o pacote é descartado.

3. Explique a função do NAT e quando ele é usado.

O recurso NAT (*Network Address Translation*) permite que os hosts de uma rede interna, que usa IPs privados, tenham acesso à internet como se estivessem diretamente conectados a ela, com o IP público usado no gateway.



109.1 Lição 2

Certificação:	LPIC-1
Versão:	5.0
Tópico:	109 Fundamentos de rede
Objetivo:	109.1 Fundamentos dos protocolos de internet
Lição:	2 de 2

Introdução

No início deste subtópico, vimos que a pilha TCP/IP é composta por uma série de protocolos diferentes. Em seguida, estudamos o protocolo IP, que permite a comunicação entre máquinas através de endereços IP, máscaras, rotas, etc.

Para que um host possa acessar um serviço disponível em outro host, além do protocolo de endereçamento IP na camada de rede, será necessário utilizar um protocolo na camada de transporte, como os protocolos TCP e UDP.

Esses protocolos realizam essa comunicação por meio de portas de rede. Portanto, além de definir um IP de origem e destino, usamos portas de origem e destino para acessar um serviço.

A porta é identificada por um campo de 16 bits, o que resulta em um limite de 65.535 portas possíveis. Os serviços (destino) usam as portas 1 a 1023, que são chamadas de *portas privilegiadas* por terem acesso de root ao sistema. A origem da conexão usa o intervalo de portas de 1024 a 65.535, chamadas de *portas não privilegiadas* ou portas de socket.

As portas usadas por cada tipo de serviço são padronizadas e controladas pela IANA (*Internet Assigned Numbers Authority*). Assim, em qualquer sistema, a porta 22 é usada pelo serviço SSH, a

porta 80 pelo serviço HTTP e assim por diante.

A tabela a seguir contém os principais serviços e suas respectivas portas.

Porta	Serviço
20	FTP (dados)
21	FTP (controle)
22	SSH (Secure Socket Shell)
23	Telnet (Conexão remota sem criptografia)
25	SMTP (Simple Mail Transfer Protocol), Enviar emails
53	DNS (Domain Name System)
80	HTTP (Hypertext Transfer Protocol)
110	POP3 (Post Office Protocol), Receber emails
123	NTP (Network Time Protocol)
139	Netbios
143	IMAP (Internet Message Access Protocol), Acessar emails
161	SNMP (Simple Network Management Protocol)
162	SNMPTRAP, Notificações SNMP
389	LDAP (Lightweight Directory Access Protocol)
443	HTTPS (HTTP Seguro)
465	SMTPS (SMTP Seguro)
514	RSH (Shell Remoto)
636	LDAPS (LDAP Seguro)
993	IMAPS (IMAP Seguro)
995	POP3S (POP3 Seguro)

Em um sistema Linux, as portas de serviço padrão são listadas no arquivo `/etc/services`.

A identificação da porta de destino desejada em uma conexão é feita usando o caractere : (dois pontos) após o endereço IPv4. Assim, ao buscar acesso ao serviço HTTPS que é servido pelo hospedeiro IP 200.216.10.15, o cliente deve enviar a solicitação para o destino

200.216.10.15:443.

Os serviços listados acima, e todos os outros, utilizam um protocolo de transporte de acordo com as características exigidas pelo serviço, sendo o TCP e o UDP os principais.

Transmission Control Protocol (TCP)

O TCP (Protocolo de controle de transmissão) é um protocolo de transporte orientado a conexões. Isso significa que uma conexão é estabelecida entre o cliente, por meio da porta de socket, e o serviço, por meio da porta padrão do serviço. O protocolo é responsável por garantir que todos os pacotes sejam entregues corretamente, verificando a integridade e a ordem dos pacotes, incluindo a retransmissão de pacotes perdidos por erros de rede.

Assim, o aplicativo não precisa implementar esse controle de fluxo de dados, pois ele já é garantido pelo protocolo TCP.

User Datagram Protocol (UDP)

O UDP (Protocolo de datagrama do usuário) estabelece uma conexão entre o cliente e o serviço, mas não controla a transmissão de dados dessa conexão. Ou seja, ele não verifica se os pacotes se perderam, se estão fora de ordem, etc. O aplicativo é responsável por implementar os controles necessários.

Como há menos controle, o UDP permite um melhor desempenho no fluxo de dados, o que é importante para alguns tipos de serviços.

Internet Control Message Protocol (ICMP)

O ICMP (Protocolo de mensagens de controle da internet) é um protocolo da camada de rede na pilha TCP/IP e sua principal função é analisar e controlar os elementos da rede, possibilitando, por exemplo:

- Controle de volume de tráfego
- Detecção de destinos inacessíveis
- Redirecionamento de rota
- Verificar o status de hosts remotos

É o protocolo utilizado pelo comando ping, que será estudado em outro subtópico.

IPv6

Até agora, estudamos a versão 4 do protocolo IP, ou seja, o IPv4. Esta tem sido a versão padrão usada em todos os ambientes de rede e internet. Porém, ela apresenta limitações, principalmente no que diz respeito ao número de endereços disponíveis, e como parece claro que em um futuro próximo todos os dispositivos estarão de alguma forma conectados à Internet (ver IoT), é cada vez mais comum utilizar a versão 6 do protocolo IP, abreviada como IPv6.

O IPv6 traz uma série de mudanças, novas implementações e recursos, bem como uma nova representação do próprio endereço.

Cada endereço IPv6 possui 128 bits, divididos em 8 grupos de 16 bits, representados por valores hexadecimais.

Por exemplo:

```
2001:0db8:85a3:08d3:1319:8a2e:0370:7344
```

Abreviações

O IPv6 define maneiras de encurtar os endereços em algumas situações. Vamos analisar o seguinte endereço:

```
2001:0db8:85a3:0000:0000:0000:0000:7344
```

A primeira possibilidade é reduzir as strings de `0000` para apenas um `0`, resultando em:

```
2001:0db8:85a3:0:0:0:0:7344
```

Além disso, no caso de grupos de strings com valor `0`, elas podem ser omitidas da seguinte forma:

```
2001:0db8:85a3::7344
```

Porém, esta última abreviatura só pode ser feita uma vez no endereço. Veja o exemplo:

```
2001:0db8:85a3:0000:0000:1319:0000:7344
```

```
2001:0db8:85a3:0:0:1319:0:7344
```

2001:0db8:85a3::1319:0:7344

Tipos de endereço IPv6

O IPv6 classifica os endereços em 3 tipos:

Unicast

Identifica uma única interface de rede. Por padrão, os 64 bits à esquerda identificam a rede e os 64 bits à direita identificam a interface.

Multicast

Identifica um conjunto de interfaces de rede. Um pacote enviado a um endereço multicast será enviado a todas as interfaces que pertencem àquele grupo. Embora semelhante, não deve ser confundido com o broadcast (transmissão), que não existe no protocolo IPv6.

Anycast

Também identifica um conjunto de interfaces na rede, mas o pacote encaminhado para um endereço *anycast* será entregue a apenas um endereço nesse conjunto, não a todos.

Diferenças entre IPv4 e IPv6

Além do endereço, diversas outras diferenças podem ser apontadas entre as versões 4 e 6 do IP. Eis algumas delas:

- As portas de serviço seguem os mesmos padrões e protocolos (TCP, UDP); a diferença está apenas na representação do IP e no conjunto de portas. No IPv6, o endereço IP deve ser protegido com [] (colchetes):

IPv4

200.216.10.15:443

IPv6

[2001:0db8:85a3:08d3:1319:8a2e:0370:7344]:443

- O IPv6 não implementa o recurso de broadcast (transmissão) da mesma maneira que o IPv4. Porém, o mesmo resultado pode ser obtido enviando-se o pacote para o endereço ff02::1, que alcança todos os hosts da rede local—semelhante a usar 224.0.0.1 no IPv4 para uma transmissão multicast.
- Por meio do recurso SLAAC (*Stateless Address Autoconfiguration*), os hosts IPv6 podem se autoconfigurar.

- O campo TTL (*Time to Live*) do IPv4 foi substituído pelo “Hop Limit” (limite de saltos) no cabeçalho do IPv6.
- Todas as interfaces IPv6 têm um endereço local, denominado endereço de link local, prefixado com `fe80::/10`.
- O IPv6 implementa o *Neighbor Discovery Protocol* (NDP), semelhante ao ARP usado pelo IPv4, mas com muito mais funcionalidades.

Exercícios Guiados

1. Qual porta é o padrão para o protocolo SMTP?

2. Quantas portas diferentes estão disponíveis em um sistema?

3. Qual protocolo de transporte garante que todos os pacotes sejam entregues corretamente, verificando a integridade e a ordem dos pacotes?

4. Que tipo de endereço IPv6 é usado para enviar um pacote a todas as interfaces que pertencem a um grupo de hosts?

Exercícios Exploratórios

1. Dê 4 exemplos de serviços que usam o protocolo TCP por padrão.

2. Qual é o nome do campo no pacote de cabeçalho do IPv6 que implementa o mesmo recurso do TTL no IPv4?

3. Que tipo de informação o Neighbour Discovery Protocol (NDP) pode descobrir?

Resumo

Esta lição cobriu os principais protocolos de transporte e serviços usados na pilha TCP/IP.

Outro tópico importante é a versão 6 do protocolo IP, incluindo os endereços IPv6 e as principais diferenças em relação ao IPv4.

Os seguintes assuntos foram abordados:

- A correlação entre números de portas e serviços
- TCP (Transmission Control Protocol)
- UDP (User Datagram Protocol)
- ICMP (Internet Control Message Protocol)
- O endereço IPv6 e como abreviá-lo
- Tipos de endereço IPv6
- Principais diferenças entre IPv4 e IPv6

Respostas aos Exercícios Guiados

1. Qual porta é o padrão para o protocolo SMTP?

25

2. Quantas portas diferentes estão disponíveis em um sistema?

65535

3. Qual protocolo de transporte garante que todos os pacotes sejam entregues corretamente, verificando a integridade e a ordem dos pacotes?

TCP

4. Que tipo de endereço IPv6 é usado para enviar um pacote a todas as interfaces que pertencem a um grupo de hosts?

Multicast

Respostas aos Exercícios Exploratórios

1. Dê 4 exemplos de serviços que usam o protocolo TCP por padrão.

FTP, SMTP, HTTP, POP3, IMAP, SSH

2. Qual é o nome do campo no pacote de cabeçalho do IPv6 que implementa o mesmo recurso do TTL no IPv4?

Hop Limit (Limite de saltos)

3. Que tipo de informação o Neighbour Discovery Protocol (NDP) pode descobrir?

O NDP é capaz de obter diversas informações da rede, incluindo outros nós, endereços duplicados, rotas, servidores DNS, gateways, etc.



Linux
Professional
Institute

109.2 Configuração persistente de rede

Referência ao LPI objectivo

LPIC-1 version 5.0, Exam 102, Objective 109.2

Peso

4

Áreas chave de conhecimento

- Configuração básica de um host TCP/IP.
- Configurar a ethernet e a rede wi-fi usando o NetworkManager.
- Noções do systemd-networkd.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- /etc/hostname
- /etc/hosts
- /etc/nsswitch.conf
- /etc/resolv.conf
- nmcli
- hostnamectl
- ifup
- ifdown



Linux
Professional
Institute

109.2 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	109 Fundamentos de rede
Objetivo:	109.2 Configuração de rede persistente
Lição:	1 de 2

Introdução

Em qualquer rede TCP/IP, cada nó deve configurar seu adaptador de rede para atender aos requisitos da rede. Caso contrário, eles não poderão se comunicar uns com os outros. Portanto, o administrador do sistema deve fornecer a configuração básica para que o sistema operacional seja capaz de configurar a interface de rede apropriada, além de se identificar e identificar os recursos básicos da rede sempre que for inicializado.

As configurações de rede não são dependentes do sistema operacional, mas cada um deles tem seus próprios métodos para armazenar e aplicar essas configurações. Os sistemas Linux dependem de configurações armazenadas em arquivos de texto simples no diretório `/etc` para ativar a conectividade de rede durante a inicialização. Vale a pena saber como esses arquivos são usados para evitar a perda de conectividade devido a uma configuração local incorreta.

A Interface de Rede

Interface de rede é o termo pelo qual o sistema operacional se refere ao canal de comunicação configurado para funcionar com o hardware de rede conectado ao sistema, como um dispositivo ethernet ou wi-fi. A exceção é a interface *loopback*, usada quando o sistema operacional precisa

estabelecer uma conexão consigo mesmo, mas o objetivo principal de uma interface de rede é fornecer uma rota através da qual os dados locais podem ser enviados e os dados remotos, recebidos. Se a interface de rede não estiver configurada corretamente, o sistema operacional não será capaz de se comunicar com outras máquinas na rede.

Na maioria dos casos, as configurações de interface corretas são definidas por padrão ou personalizadas durante a instalação do sistema operacional. Ainda assim, essas configurações geralmente precisam ser inspecionadas ou mesmo modificadas quando a comunicação não está funcionando corretamente ou quando o comportamento da interface requer personalização.

O Linux tem muitos comandos para listar as interfaces de rede presentes no sistema, mas nem todos estão disponíveis em todas as distribuições. O comando `ip`, no entanto, faz parte do conjunto básico de ferramentas de rede empacotadas com todas as distribuições Linux e pode ser usado para listar as interfaces de rede. O comando completo para mostrar as interfaces é `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
```

Se disponível, o comando `nmcli device` também pode ser usado:

```
$ nmcli device
DEVICE      TYPE      STATE      CONNECTION
enp3s5      ethernet  connected  Gigabit Powerline Adapter
lo          loopback  unmanaged  --
```

Os comandos mostrados nos exemplos não modificam as configurações do sistema e, portanto, podem ser executados por um usuário sem privilégios. Ambos os comandos listam duas interfaces de rede: `lo` (a interface de loopback) e `enp3s5` (uma interface ethernet).

Os desktops e laptops que executam Linux geralmente têm duas ou três interfaces de rede predefinidas, uma para a interface virtual de loopback e as outras atribuídas ao hardware de rede encontrado pelo sistema. Os servidores e dispositivos de rede que executam Linux, por outro lado, podem ter dezenas de interfaces de rede, mas os mesmos princípios se aplicam a todas elas. A abstração fornecida pelo sistema operacional permite a configuração de interfaces de rede

usando os mesmos métodos, independentemente do hardware usado.

No entanto, é útil conhecer os detalhes sobre o hardware subjacente de uma interface para entender melhor o que está acontecendo quando a comunicação não funciona conforme o esperado. Em um sistema em que há muitas interfaces de rede disponíveis, nem sempre é óbvio saber qual delas corresponde ao wi-fi e qual corresponde à ethernet, por exemplo. Por esse motivo, o Linux usa uma convenção para a nomenclatura de interfaces que ajuda a identificar qual interface de rede corresponde a qual dispositivo e porta.

Nomes de interface

As distribuições Linux mais antigas atribuíam às interfaces de rede ethernet nomes como `eth0`, `eth1`, etc., numeradas de acordo com a ordem em que o kernel identificava os dispositivos. As interfaces wireless eram nomeadas `wlan0`, `wlan1`, etc. Esta convenção de nomenclatura, no entanto, não esclarece qual porta Ethernet específica corresponde à interface `eth0`, por exemplo. Dependendo de como o hardware fosse detectado, era possível inclusive que duas interfaces de rede trocassem de nome após uma reinicialização.

Para evitar essa ambigüidade, os sistemas Linux mais recentes empregam uma convenção de nomenclatura previsível para as interfaces de rede, criando uma aproximação maior entre o nome da interface e a conexão de hardware subjacente.

Nas distribuições Linux que usam o esquema de nomenclatura do systemd, todos os nomes de interfaces começam com um prefixo de dois caracteres que indica o tipo de interface:

en

Ethernet

ib

InfiniBand

s1

Serial line IP (slip)

wl

Rede de área local sem fio (WLAN)

ww

Rede de longa distância sem fio (WWAN)

De prioridade mais alta para mais baixa, as seguintes regras são usadas pelo sistema operacional

para nomear e numerar as interfaces de rede:

1. Nomear a interface de acordo com o índice fornecido pela BIOS ou pelo firmware dos dispositivos incorporados, por exemplo, `eno1`.
2. Nomear a interface de acordo com o índice do slot PCI Express, conforme fornecido pela BIOS ou firmware, por exemplo, `ens1`.
3. Nomear a interface de acordo com seu endereço no barramento correspondente, por exemplo, `enp3s5`.
4. Nomear a interface de acordo com o endereço MAC da interface, por exemplo, `enx78e7d1ea46da`.
5. Nomear a interface usando a convenção legada, por exemplo, `eth0`.

É correto pressupor, por exemplo, que a interface de rede `enp3s5` recebeu esse nome porque não se encaixava nos dois primeiros métodos de nomenclatura, de forma que seu endereço no barramento e no slot correspondentes foi usado. O endereço do dispositivo `03:05.0`, encontrado na saída do comando `lspci`, revela o dispositivo associado:

```
$ lspci | grep Ethernet
03:05.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL-8110SC/8169SC Gigabit
Ethernet (rev 10)
```

As interfaces de rede são criadas pelo próprio kernel do Linux, mas existem muitos comandos que podem ser usados para interagir com elas. Normalmente, a configuração ocorre automaticamente e não há necessidade de se alterar as configurações manualmente. Ainda assim, com o nome da interface, é possível informar ao kernel como proceder para configurá-la se necessário.

Gerenciamento da interface

Ao longo dos anos, vários programas foram desenvolvidos para interagir com os recursos de rede fornecidos pelo kernel do Linux. Embora o antigo comando `ifconfig` ainda possa ser usado para fazer consultas e configurações de interface simples, ele agora está obsoleto devido ao seu suporte limitado a interfaces não ethernet. O `ifconfig` foi substituído pelo comando `ip`, que é capaz de gerenciar muitos outros aspectos das interfaces TCP/IP, como rotas e túneis.

Para a maioria das tarefas comuns, a riqueza de recursos do comando `ip` acaba sendo excessiva, e por isso existem comandos auxiliares que facilitam a ativação e configuração das interfaces de rede. Os comandos `ifup` e `ifdown` servem para configurar interfaces de rede com base nas definições de interface encontradas no arquivo `/etc/network/interfaces`. Embora possam ser invocados manualmente, esses comandos são, em geral, executados automaticamente durante a

inicialização do sistema.

Todas as interfaces de rede gerenciadas por `ifup` e `ifdown` devem estar listadas no arquivo `/etc/network/interfaces`. O formato usado no arquivo é simples: as linhas que começam com a palavra `auto` são usadas para identificar as interfaces físicas a serem acessadas quando o `ifup` é executado com a opção `-a`. O nome da interface deve seguir a palavra `auto` na mesma linha. Todas as interfaces marcadas como `auto` são ativadas no momento da inicialização, na ordem em que estão listadas.

WARNING

Os métodos de configuração de rede usados por `ifup` e `ifdown` não são padronizados em todas as distribuições Linux. O CentOS, por exemplo, mantém as configurações de interface em arquivos individuais no diretório `/etc/sysconfig/network-scripts/`, e o formato de configuração usado neles é ligeiramente diferente do formato usado em `/etc/network/interfaces`.

A configuração da interface em si é escrita em outra linha, começando com a palavra `iface`, seguida pelo nome da interface, o nome da família de endereços usada por ela e o nome do método empregado para configurá-la. O exemplo a seguir mostra um arquivo de configuração básico para as interfaces `lo` (loopback) e `enp3s5`:

```
auto lo
iface lo inet loopback

auto enp3s5
iface enp3s5 inet dhcp
```

A família de endereços deve ser `inet` para as redes TCP/IP, mas também há suporte a redes IPX (`ipx`) e IPv6 (`inet6`). As interfaces de loopback usam o método de configuração `loopback`. Com o método `dhcp`, a interface usa as configurações de IP fornecidas pelo servidor DHCP da rede. As configurações de nosso exemplo permitem a execução do comando `ifup` usando o nome de interface `enp3s5` como argumento:

```
# 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
DHCPCDISCOVER 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.
```

Neste exemplo, o método escolhido para a interface `enp3s5` foi `dhcp`, de modo que o comando `ifup` chamou um programa cliente DHCP para obter as configurações de IP do servidor DHCP. Da mesma forma, o comando `ifdown enp3s5` pode ser usado para desativar a interface.

Nas redes sem um servidor DHCP, o método `static` pode ser usado e as configurações de IP fornecidas manualmente em `/etc/network/interfaces`. Por exemplo:

```
iface enp3s5 inet static
    address 192.168.1.2/24
    gateway 192.168.1.1
```

As interfaces que usam o método `static` não precisam de uma instrução `auto` correspondente, pois são ativadas sempre que o hardware de rede é detectado.

Se a mesma interface tiver mais de uma entrada `iface`, todos os endereços e opções configurados serão aplicados ao se abrir essa interface. Isso é útil para configurar endereços IPv4 e IPv6 na mesma interface, bem como para configurar diversos endereços do mesmo tipo em uma única interface.

Nomes locais e remotos

Uma configuração funcional de TCP/IP é apenas o primeiro passo para a usabilidade total da rede. Além de ser capaz de distinguir os nós da rede por seus números IP, o sistema deve poder identificá-los com nomes mais facilmente compreensíveis por seres humanos.

O nome pelo qual o sistema se identifica é personalizável e é aconselhável fazê-lo, mesmo se a máquina não for destinada a se conectar a uma rede. O nome local geralmente corresponde ao nome da rede da máquina, mas nem sempre. Se o arquivo `/etc/hostname` existir, o sistema operacional usará o conteúdo da primeira linha como nome local, que a partir daí é chamado simplesmente de *nome do host* (`hostname`, em inglês). As linhas que começam com `#` dentro de `/etc/hostname` são ignoradas.

O arquivo `/etc/hostname` pode ser editado diretamente, mas o nome de host da máquina também pode ser definido com o comando `hostnamectl`. Quando fornecido com o subcomando

`set-hostname`, o comando `hostnamectl` pega o nome dado como argumento e o escreve em `/etc/hostname`:

```
# hostnamectl set-hostname storage
# cat /etc/hostname
storage
```

O nome de host (ou “hospedeiro”) definido em `/etc/hostname` é o nome *estático*, ou seja, o nome usado para inicializar o nome de host do sistema na inicialização. O nome de host estático pode ser uma string com até 64 caracteres de comprimento. No entanto, é recomendado que ele consista apenas em caracteres ASCII minúsculos e sem espaços ou pontos. Também é aconselhável limitá-lo ao formato permitido para rótulos de nomes de domínio DNS, embora esse não seja um requisito estrito.

O comando `hostnamectl` pode definir dois outros tipos de nomes de host além do nome de host estático:

Hostname pretty

Ao contrário do nome de host estático, o nome de host pretty pode incluir todos os tipos de caracteres especiais. Ele pode ser usado para definir um nome mais descriptivo para a máquina, por exemplo “Armazenamento compartilhado da LAN”:

```
# hostnamectl --pretty set-hostname "LAN Shared Storage"
```

Hostname transiente

Usado quando o nome de host estático não está definido ou quando ele é o nome `localhost` padrão. O nome de host transiente é normalmente definido junto com outras configurações automáticas, mas também pode ser modificado com o comando `hostnamectl`, por exemplo:

```
# hostnamectl --transient set-hostname generic-host
```

Se nem a opção `--pretty` nem `--transient` forem usadas, os três tipos de nomes de host serão configurados com o nome fornecido. Para definir o nome de host estático, mas não os nomes `pretty` e `transiente`, usa-se a opção `--static`. Em todos os casos, somente o nome de host estático é armazenado no arquivo `/etc/hostname`. O comando `hostnamectl` também pode ser usado para exibir diversas informações descriptivas e de identidade sobre o sistema em execução:

```
$ 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
```

Esta é a ação padrão do comando `hostnamectl`, de modo que o subcomando `status` pode ser omitido.

Com relação ao nome dos nós remotos da rede, o sistema operacional tem duas maneiras básicas à sua disposição para combinar nomes e números IP: usar uma fonte local ou usar um servidor remoto para traduzir nomes em números IP e vice-versa. Os métodos podem ser complementares entre si e sua ordem de prioridade é definida no arquivo de configuração *Name Service Switch*: `/etc/nsswitch.conf`. Este arquivo é usado pelo sistema e pelos aplicativos para determinar não apenas as fontes para as correspondências de nome-IP, mas também as fontes das quais obter informações de nome-serviço em uma variedade de categorias, chamadas *bancos de dados*.

O banco de dados *hosts* mantém um registro do mapeamento entre nomes e números de host. A linha dentro de `/etc/nsswitch.conf` que começa com `hosts` define os serviços responsáveis por fornecer as associações para ele:

```
hosts: files dns
```

Neste exemplo, `files` e `dns` são os nomes de serviços que especificam como o processo de pesquisa de nomes de host funciona. Primeiro, o sistema procura por correspondências em arquivos locais e, em seguida, solicita correspondências ao serviço DNS.

O arquivo local do banco de dados *hosts* é `/etc/hosts`, um arquivo de texto simples que associa endereços IP a nomes de host, com uma linha por endereço IP, por exemplo:

```
127.0.0.1 localhost
```

O número IP `127.0.0.1` é o endereço padrão da interface de loopback, daí sua associação com o nome `localhost`.

Também é possível vincular aliases opcionais ao mesmo IP. Os aliases podem fornecer grafias

alternativas, nomes de host mais curtos, e devem ser adicionados no final da linha, por exemplo:

```
192.168.1.10 foo.mydomain.org foo
```

As regras de formatação para o arquivo `/etc/hosts` são:

- Os campos da entrada são separados por qualquer número de espaços em branco e/ou caracteres de tabulação.
- O texto que começa em um caractere `#` e vai até o final da linha é um comentário e é ignorado.
- Os nomes de host podem conter apenas caracteres alfanuméricos, sinais de menos e pontos.
- Os nomes de host devem começar com um caractere alfabético e terminar com um caractere alfanumérico.

Também podemos adicionar endereços IPv6 a `/etc/hosts`. A seguinte entrada se refere ao endereço de loopback IPv6:

```
::1 localhost ip6-localhost ip6-loopback
```

Após a especificação do serviço `files`, a especificação `dns` diz ao sistema para solicitar a um serviço DNS a associação nome/IP desejada. O conjunto de rotinas responsáveis por este método é chamado de *resolver* (resolvedor) e seu arquivo de configuração é `/etc/resolv.conf`. O exemplo a seguir mostra um `/etc/resolv.conf` genérico contendo entradas para os servidores DNS públicos do Google:

```
nameserver 8.8.4.4
nameserver 8.8.8.8
```

Como mostrado no exemplo, a palavra-chave `nameserver` indica o endereço IP do servidor DNS. Apenas um servidor de nomes é necessário, mas é possível informar até três deles. Os suplementares serão usados como reserva. Se nenhuma entrada de servidor de nomes estiver presente, o comportamento padrão é usar o servidor de nomes da máquina local.

O resolver pode ser configurado para adicionar automaticamente o domínio aos nomes antes de consultá-los no servidor de nomes. Por exemplo:

```
nameserver 8.8.4.4
nameserver 8.8.8.8
domain mydomain.org
```

```
search mydomain.net mydomain.com
```

A entrada `domain` define `mydomain.org` como o nome de domínio local; assim, as consultas por nomes dentro deste domínio podem usar nomes curtos relativos ao domínio local. A entrada `search` tem uma finalidade semelhante, mas aceita uma lista de domínios a experimentar quando um nome curto é fornecido. Por padrão, ela contém apenas o nome de domínio local.

Exercícios Guiados

1. Quais comandos podem ser usados para listar os adaptadores de rede presentes no sistema?

2. Qual é o tipo de adaptador de rede cujo nome de interface é `wlo1`?

3. Qual o papel do arquivo `/etc/network/interfaces` durante o tempo de inicialização?

4. Qual entrada de `/etc/network/interfaces` configura a interface `eno1` para obter suas configurações de IP com DHCP?

Exercícios Exploratórios

1. Como o comando `hostnamectl` poderia ser usado para alterar apenas o nome de host *estático* da máquina local para `firewall`?

2. Quais detalhes além dos nomes de host podem ser modificados pelo comando `hostnamectl`?

3. Qual entrada de `/etc/hosts` associa os nomes `firewall` e `router` com o IP `10.8.0.1`?

4. Como o arquivo `/etc/resolv.conf` poderia ser modificado de maneira a enviar todas as solicitações de DNS para `1.1.1.1`?

Resumo

Esta lição trata de como fazer mudanças persistentes na configuração da rede local usando arquivos e comandos padrão do Linux. O Linux espera que as configurações de TCP/IP estejam em locais específicos e pode ser necessário alterá-las quando as configurações padrão não forem apropriadas. A lição abrange os seguintes tópicos:

- Como o Linux identifica as interfaces de rede.
- Ativação da interface durante a inicialização e configuração de IP básica.
- Como o sistema operacional associa nomes a hosts.

Os conceitos, comandos e procedimentos abordados foram:

- Convenções de nomenclatura de interfaces.
- Listagem de interfaces de rede com `ip` e `nmcli`.
- Ativação da interface com `ifup` e `ifdown`.
- Comando `hostnamectl` e o arquivo `/etc/hostname`.
- Arquivos `/etc/nsswitch.conf`, `/etc/hosts` e `/etc/resolv.conf`.

Respostas aos Exercícios Guiados

1. Quais comandos podem ser usados para listar os adaptadores de rede presentes no sistema?

Os comandos `ip link show`, `nmcli device` e o comando legado `ifconfig`.

2. Qual é o tipo de adaptador de rede cujo nome de interface é `wlo1`?

O nome começa com `wl`, portanto trata-se de um adaptador LAN sem fio.

3. Qual o papel do arquivo `/etc/network/interfaces` durante o tempo de inicialização?

Ele contém as configurações utilizadas pelo comando `ifup` para ativar as interfaces correspondentes durante o tempo de inicialização.

4. Qual entrada de `/etc/network/interfaces` configura a interface `eno1` para obter suas configurações de IP com DHCP?

A linha `iface eno1 inet dhcp`.

Respostas aos Exercícios Exploratórios

1. Como o comando `hostnamectl` poderia ser usado para alterar apenas o nome de host *estático* da máquina local para `firewall`?

Com a opção `--static`: `hostnamectl --static set-hostname firewall`.

2. Quais detalhes além dos nomes de host podem ser modificados pelo comando `hostnamectl`?

O `hostnamectl` também pode definir o ícone padrão da máquina local, o tipo de chassis, a localização e o ambiente de implantação.

3. Qual entrada de `/etc/hosts` associa os nomes `firewall` e `router` com o IP `10.8.0.1`?

A linha `10.8.0.1 firewall router`.

4. Como o arquivo `/etc/resolv.conf` poderia ser modificado de maneira a enviar todas as solicitações de DNS para `1.1.1.1`?

Usando `nameserver 1.1.1.1` como sua única entrada de nameserver.



**Linux
Professional
Institute**

109.2 Lição 2

Certificação:	LPIC-1
Versão:	5.0
Tópico:	109 Fundamentos de rede
Objetivo:	109.2 Configuração de rede persistente
Lição:	2 de 2

Introdução

O Linux oferece suporte a praticamente todas as tecnologias de rede usadas para conectar servidores, contêineres, máquinas virtuais, desktops e dispositivos móveis. As conexões entre todos esses nós da rede podem ser dinâmicas e heterogêneas, exigindo, assim, um gerenciamento adequado por parte do sistema operacional instalado.

No passado, as distribuições desenvolviam suas próprias soluções personalizadas para gerenciar a infraestrutura de rede dinâmica. Hoje, ferramentas como o *NetworkManager* e o *systemd* fornecem recursos mais abrangentes e integrados para atender a todas as demandas específicas.

NetworkManager

A maioria das distribuições Linux adota o daemon do serviço *NetworkManager* para configurar e controlar as conexões de rede do sistema. A finalidade do *NetworkManager* é tornar a configuração da rede o mais simples e automática possível. Ao usarmos o DHCP, por exemplo, o *NetworkManager* organiza as mudanças de rota, a busca de endereço IP e atualizações da lista local de servidores DNS, se necessário. Quando há conexões com e sem fio disponíveis, o *NetworkManager* prioriza a conexão com fio por padrão. O *NetworkManager* tenta manter ao

menos uma conexão ativa o tempo todo, sempre que possível.

NOTE

Uma solicitação usando DHCP (*Dynamic Host Configuration Protocol*, ou Protocolo de configuração de host dinâmico) geralmente é enviada por meio do adaptador de rede assim que o link para a rede é estabelecido. O servidor DHCP que está ativo na rede responde com as configurações (endereço IP, máscara de rede, rota padrão, etc.) que o solicitante deve usar para se comunicar por meio do protocolo IP.

Por padrão, o daemon do NetworkManager controla as interfaces de rede não mencionadas no arquivo `/etc/network/interfaces`. Ele faz isso para não interferir em outros métodos de configuração que também podem estar presentes, modificando assim somente as interfaces desaccompanhadas.

O serviço NetworkManager é executado em segundo plano com privilégios de root e aciona as ações necessárias para manter o sistema online. Os usuários comuns podem criar e modificar conexões de rede com aplicativos clientes que, mesmo sem privilégios de root, são capazes de se comunicar com o serviço subjacente para executar as ações solicitadas.

Existem aplicativos cliente do NetworkManager para a linha de comando e para o ambiente gráfico. No caso deste último, o aplicativo cliente é incluído como um acessório do ambiente de desktop (com nomes como `nm-tray`, `network-manager-gnome`, `nm-applet` ou `plasma-nm`) e geralmente fica acessível por meio de um ícone indicador no canto da barra da área de trabalho ou no utilitário de configuração do sistema.

Na linha de comando, o próprio NetworkManager fornece dois programas clientes: `nmcli` e `nmtui`. Ambos trazem os mesmos recursos básicos, mas o `nmtui` tem uma interface baseada em curses, ao passo que o `nmcli` é um comando mais abrangente que também pode ser usado em scripts. O comando `nmcli` separa todas as propriedades relacionadas à rede controladas pelo NetworkManager em categorias chamadas *objetos*:

general

Status e operações gerais do NetworkManager

networking

Controle geral de rede.

radio

Controles de rádio do NetworkManager.

connection

Coneções do NetworkManager.

device

Dispositivos gerenciados pelo NetworkManager.

agent

Agente secreto ou agente polkit do NetworkManager.

monitor

Monitora as mudanças do NetworkManager.

O nome do objeto é o principal argumento do comando `nmcli`. Para mostrar o status geral de conectividade do sistema, por exemplo, o objeto `general` deve ser dado como argumento:

```
$ nmcli general
STATE      CONNECTIVITY  WIFI-HW  WIFI      WWAN-HW  WWAN
connected   full        enabled   enabled   enabled   enabled
```

A coluna `STATE` informa se o sistema está conectado a uma rede ou não. Se a conexão for limitada devido a uma configuração externa incorreta ou a restrições de acesso, a coluna `CONNECTIVITY` não relatará um status de conectividade `full`. Se `Portal` aparecer na coluna `CONNECTIVITY`, isso significa que são necessárias etapas extras de autenticação (geralmente por meio do navegador web) para concluir o processo de conexão. As colunas restantes mostram o status das conexões sem fio (se houver), seja `WIFI` ou `WWAN` (Wide Wireless Area Network, ou seja, redes celulares). O sufixo `HW` indica que o status corresponde ao dispositivo de rede e não à conexão de rede do sistema, ou seja, informa se o hardware está habilitado ou desabilitado para economizar energia.

Além do argumento de objeto, o `nmcli` também precisa de um argumento de comando para ser executado. O comando `status` é usado por padrão se nenhum argumento de comando estiver presente, de modo que o comando `nmcli general` é interpretado, na verdade, como `nmcli general status`.

Praticamente não é necessário realizar nenhuma ação quando o adaptador de rede está conectado diretamente ao ponto de acesso por meio de cabos, mas as redes sem fio exigem mais interação para aceitar novos membros. O `nmcli` facilita o processo de conexão e salva as configurações para permitir uma conexão automática no futuro e, portanto, é muito útil para laptops ou qualquer outro dispositivo móvel.

Antes de conectar-se ao wi-fi, é conveniente listar primeiro as redes disponíveis na área local. Se o sistema tiver um adaptador wi-fi funcionando, o objeto `device` irá usá-lo para verificar as redes disponíveis com o comando `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

A maioria dos usuários provavelmente usará o nome na coluna `SSID` para identificar a rede de interesse. Por exemplo, o comando `nmcli` pode se conectar à rede de nome `Hypnotoad` usando o objeto `device` novamente:

```
$ nmcli device wifi connect Hypnotoad
```

Se o comando for executado dentro de um emulador de terminal no ambiente gráfico, aparecerá uma caixa de diálogo solicitando a senha da rede. Quando executado em um console de texto, a senha pode ser fornecida junto com os outros argumentos:

```
$ nmcli device wifi connect Hypnotoad password MyPassword
```

Se a rede wi-fi esconde seu nome `SSID`, o `nmcli` ainda assim pode se conectar a ela com os argumentos extras `hidden yes`:

```
$ nmcli device wifi connect Hypnotoad password MyPassword hidden yes
```

Se o sistema tiver mais de um adaptador wi-fi, indicamos o que deve ser usado com `ifname`. Por exemplo, para se conectar usando o adaptador de nome `wlo1`:

```
$ nmcli device wifi connect Hypnotoad password MyPassword ifname wlo1
```

Depois que a conexão for estabelecida, o NetworkManager dará um nome a ela de acordo com o `SSID` correspondente (no caso de uma conexão wi-fi) e manterá esse nome nas conexões futuras. O nome das conexões e seus UIDs são listados pelo comando `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	--

O tipo de cada conexão é mostrado—pode ser `ethernet`, `wifi`, `tun`, `gsm`, `bridge`, etc.—bem como o dispositivo ao qual estão associadas. Para executar ações em uma conexão específica, é preciso fornecer seu nome ou UUID. Para desativar a conexão Hypnotoad, por exemplo:

```
$ nmcli connection down Hypnotoad
Connection 'Hypnotoad' successfully deactivated
```

Da mesma forma, o comando `nmcli connection up Hypnotoad` pode ser usado para ativar a conexão, pois agora ela está salva pelo NetworkManager. Também é possível se reconectar com o nome da interface, mas, nesse caso, usamos o objeto `device`:

```
$ nmcli device disconnect wlo2
Device 'wlo1' successfully disconnected.
```

O nome da interface também pode ser usado para restabelecer a conexão:

```
$ nmcli device connect wlo2
Device 'wlo1' successfully activated with '833692de-377e-4f91-a3dc-d9a2b1fcf6cb'.
```

Observe que o UUID da conexão muda a cada vez que a conexão é ativada e, portanto, é preferível usar o nome para manter a consistência.

Se houver um adaptador sem fio disponível, mas ele não estiver sendo usado, ele pode ser desligado para economizar energia. Desta vez, o objeto `radio` deve ser passado para `nmcli`:

```
$ nmcli radio wifi off
```

Obviamente, o dispositivo sem fio pode ser ativado novamente com o comando `nmcli radio wifi on`.

Uma vez que as conexões forem estabelecidas, nenhuma interação manual será necessária no futuro, pois o NetworkManager identifica as redes conhecidas disponíveis e se conecta a elas automaticamente. Se necessário, o NetworkManager possui plugins que podem estender suas funcionalidades, por exemplo para suportar conexões VPN.

systemd-networkd

Os sistemas que rodam o systemd podem opcionalmente usar os daemons nativos para gerenciar a conectividade de rede: `systemd-networkd` para controlar as interfaces de rede e `systemd-resolution` para gerenciar a resolução de nome local. Esses serviços são compatíveis com os métodos de configuração legados do Linux, mas a configuração das interfaces de rede em particular tem recursos que vale a pena conhecer.

Os arquivos de configuração usados pelo `systemd-networkd` para configurar as interfaces de rede podem ser encontrados em um dos três diretórios a seguir:

`/lib/systemd/network`

O diretório de rede do sistema.

`/run/systemd/network`

O diretório volátil de tempo de execução da rede.

`/etc/systemd/network`

O diretório local de administração da rede.

Os arquivos são processados em ordem lexicográfica, por isso é recomendável iniciar seus nomes com números para facilitar o ordenamento e a leitura.

Os arquivos em `/etc` têm a prioridade mais alta, ao passo que os arquivos em `/run` têm precedência sobre os arquivos com o mesmo nome em `/lib`. Ou seja, se dois ou mais arquivos de configuração em diretórios diferentes tiverem o mesmo nome, o `systemd-networkd` ignora os que tiverem menor prioridade. Essa maneira de separar os arquivos permite mudar as configurações da interface sem que seja necessário modificar os arquivos originais: as modificações podem ser postas em `/etc/systemd/network` para sobrescrever as existentes em `/lib/systemd/network`.

A finalidade de cada arquivo de configuração depende de seu sufixo. Os nomes de arquivos que terminam em `.netdev` são usados pelo `systemd-networkd` para criar dispositivos de rede virtuais, como dispositivos `bridge` ou `tun`. Os arquivos que terminam em `.link` definem configurações de baixo nível para a interface de rede correspondente. O `systemd-networkd` detecta e configura os dispositivos de rede automaticamente conforme eles aparecem — além de ignorar dispositivos já configurados por outros meios — e, portanto, há pouca necessidade de adicionar esses arquivos na maioria das situações.

O sufixo mais importante é `.network`. Os arquivos que empregam esse sufixo podem ser usados para configurar endereços de rede e rotas. Tal como acontece com os outros tipos de arquivos de configuração, o nome do arquivo define a ordem em que ele será processado. A interface de rede

à qual o arquivo de configuração se refere é definida na seção [Match] ` dentro do arquivo.

Por exemplo, a interface de rede ethernet `enp3s5` pode ser selecionada dentro do arquivo `/etc/systemd/network/30-lan.network` graças à entrada `Name=enp3s5` na seção [Match]:

```
[Match]
Name=enp3s5
```

Também é possível usar uma lista de nomes separados por espaços em branco para selecionar diversas interfaces de rede de uma vez neste mesmo arquivo. Os nomes podem conter globos no estilo do shell, como `en*`. Outras entradas permitem usar regras diferentes, como por exemplo selecionar um dispositivo de rede por seu endereço MAC:

```
[Match]
MACAddress=00:16:3e:8d:2b:5b
```

As configurações do dispositivo estão na seção [Network] do arquivo. Uma configuração de rede estática simples requer apenas as entradas `Address` e `Gateway`:

```
[Match]
MACAddress=00:16:3e:8d:2b:5b

[Network]
Address=192.168.0.100/24
Gateway=192.168.0.1
```

Para usar o protocolo DHCP em vez de endereços IP estáticos, a entrada `DHCP` deve ser usada:

```
[Match]
MACAddress=00:16:3e:8d:2b:5b

[Network]
DHCP=yes
```

O serviço `systemd-networkd` tenta buscar os endereços IPv4 e IPv6 para a interface de rede. Para usar apenas IPv4, empregamos `DHCP=ipv4`. Da mesma forma, `DHCP=ipv6` ignora as configurações IPv4 e usa apenas o endereço IPv6 fornecido.

As redes sem fio protegidas por senha também podem ser configuradas pelo `systemd-networkd`,

mas o adaptador de rede já deve estar autenticado na rede antes que o systemd-networkd possa configurá-lo. A autenticação é realizada pelo *WPA supplicant*, um programa dedicado a configurar adaptadores de rede para redes protegidas por senha.

O primeiro passo é criar o arquivo de credenciais com o comando `wpa_passphrase`:

```
# wpa_passphrase MyWifi > /etc/wpa_supplicant/wpa_supplicant-wlo1.conf
```

Este comando pega a frase-senha para a rede sem fio MyWifi da entrada padrão e armazena seu hash em `/etc/wpa_supplicant/wpa_supplicant-wlo1.conf`. Note que o nome do arquivo deve conter o nome apropriado da interface sem fio, por isso o `wlo1` no nome do arquivo.

O gerenciador do systemd lê os arquivos de frase-senha do WPA em `/etc/wpa_supplicant/` e cria o serviço correspondente para executar o WPA supplicant e abrir a interface. O arquivo de frase secreta criado no exemplo terá uma unidade de serviço correspondente chamada `wpa_supplicant@wlo1.service`. O comando `systemctl start wpa_supplicant@wlo1.service` associará o adaptador sem fio ao ponto de acesso remoto. O comando `systemctl enable wpa_supplicant@wlo1.service` tornará essa associação automática durante a inicialização.

Finalmente, um arquivo `.network` correspondente à interface `wlo1` deve estar presente em `/etc/systemd/network/`, já que o systemd-networkd vai usá-lo para configurar a interface assim que o WPA supplicant encerrar a associação com o ponto de acesso.

Exercícios Guiados

1. O que significa a palavra Portal na coluna CONNECTIVITY na saída do comando `nmcli general status`?

2. Em um terminal de console, como um usuário comum pode usar o comando `nmcli` para se conectar à rede sem fio MyWifi protegida pela senha MyPassword?

3. Qual comando pode ativar o adaptador sem fio se ele tiver sido desabilitado anteriormente pelo sistema operacional?

4. Em que diretório devem ser postos os arquivos de configuração personalizados quando o `systemd-networkd` está gerenciando as interfaces de rede?

Exercícios Exploratórios

1. Como um usuário pode executar o comando `nmcli` para excluir uma conexão não utilizada chamada `Hotel Internet`?

2. O NetworkManager varre redes wi-fi periodicamente e o comando `nmcli device wifi list` lista apenas os pontos de acesso encontrados na última varredura. Como usar o comando `nmcli` para pedir que o NetworkManager verifique novamente todos os pontos de acesso disponíveis?

3. Qual entrada `name` deve ser usada na seção `[Match]` de um arquivo de configuração do `systemd-networkd` para encontrar todas as interfaces `ethernet`?

4. Como o comando `wpa_passphrase` deve ser executado para usar a frase-senha fornecida como argumento e não a partir da entrada padrão?

Resumo

Esta lição cobre as ferramentas comuns usadas no Linux para gerenciar conexões de rede heterogêneas e dinâmicas. Embora a maioria dos métodos de configuração não exija intervenção do usuário, às vezes isso é necessário, e ferramentas como o *NetworkManager* e o *systemd-networkd* podem reduzir o incômodo ao mínimo. A lição abrange os seguintes tópicos:

- Como o NetworkManager e o systemd-networkd se integram ao sistema.
- Como o usuário pode interagir com o NetworkManager e o systemd-networkd.
- Configuração básica de interfaces com o NetworkManager e o systemd-networkd.

Os conceitos, comandos e procedimentos abordados foram:

- Comandos do cliente do NetworkManager: `nmtui` and `nmcli`.
- Varredura e conexão com redes sem fio usando os comandos apropriados do `nmcli`.
- Conexões de rede wi-fi persistentes usando o systemd-networkd.

Respostas aos Exercícios Guiados

1. O que significa a palavra Portal na coluna CONNECTIVITY na saída do comando `nmcli general status`?

Significa que são necessárias etapas extras de autenticação (geralmente por meio do navegador web) para concluir o processo de conexão.

2. Em um terminal de console, como um usuário comum pode usar o comando `nmcli` para se conectar à rede sem fio MyWifi protegida pela senha MyPassword?

Em um terminal somente texto, o comando seria

```
$ nmcli device wifi connect MyWifi password MyPassword
```

3. Qual comando pode ativar o adaptador sem fio se ele tiver sido desabilitado anteriormente pelo sistema operacional?

```
$ nmcli radio wifi on
```

4. Em que diretório devem ser postos os arquivos de configuração personalizados quando o `systemd-networkd` está gerenciando as interfaces de rede?

No diretório local de administração da rede: `/etc/systemd/network`.

Respostas aos Exercícios Exploratórios

- Como um usuário pode executar o comando `nmcli` para excluir uma conexão não utilizada chamada Hotel Internet?

```
$ nmcli connection delete "Hotel Internet"
```

- O NetworkManager varre redes wi-fi periodicamente e o comando `nmcli device wifi list` lista apenas os pontos de acesso encontrados na última varredura. Como usar o comando `nmcli` para pedir que o NetworkManager verifique novamente todos os pontos de acesso disponíveis?

O usuário root pode executar `nmcli device wifi rescan` para fazer o NetworkManager verificar novamente os pontos de acesso disponíveis.

- Qual entrada `name` deve ser usada na seção `[Match]` de um arquivo de configuração do `systemd-networkd` para encontrar todas as interfaces ethernet?

A entrada `name=en*`, já que `en` é o prefixo para as interfaces ethernet no Linux e o `systemd-networkd` aceita globos do tipo shell.

- Como o comando `wpa_passphrase` deve ser executado para usar a frase-senha fornecida como argumento e não a partir da entrada padrão?

A senha deve ser fornecida logo após o SSID, como em `wpa_passphrase MyWifi MyPassword`.



109.3 Soluções para problemas simples de rede

Referência ao LPI objectivo

[LPIC-1 version 5.0, Exam 102, Objective 109.3](#)

Peso

4

Áreas chave de conhecimento

- Configuração manual de interfaces de rede, incluindo verificar e alterar a configuração de interfaces de rede usando o iproute2.
- Configuração manual de tabelas de roteamento, incluindo verificar e alterar a tabela de rotas e definir a rota padrão usando o iproute2.
- Solucionar problemas associados com a configuração da rede.
- Noções dos comandos legados do net-tools.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- `ip`
- `hostname`
- `ss`
- `ping`
- `ping6`
- `traceroute`
- `traceroute6`
- `tracepath`
- `tracepath6`

- netcat
- ifconfig
- netstat
- route



Linux
Professional
Institute

109.3 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	109 Fundamentos de rede
Objetivo:	109.3 Resolução de problemas básicos de rede
Lição:	1 de 2

Introdução

O Linux dispõe de recursos de rede extremamente flexíveis e poderosos. Na verdade, os sistemas operacionais baseados em Linux são freqüentemente usados nos dispositivos de rede comuns, incluindo equipamentos comerciais de alto padrão. Seria possível criar uma certificação inteira tendo como tema os recursos de rede do Linux. Tendo isso em mente, esta lição abordará somente algumas ferramentas básicas de configuração e resolução de problemas.

Sugerimos revisar as lições sobre protocolos de internet e configurações de rede persistente antes de iniciar esta lição. Aqui, trataremos das ferramentas para configurar e solucionar problemas nas redes IPv4 e IPv6.

Embora não estejam incluídos nos objetivos oficiais, *farejadores de pacotes* (packet sniffers) como o `tcpdump` são ferramentas úteis para a resolução de problemas. Os sniffers de pacotes permitem visualizar e gravar os pacotes que entram ou saem de uma interface de rede. Ferramentas como *visualizadores hexadecimais* e *analisadores de protocolo* podem ser usadas para visualizar esses pacotes com mais detalhes do que um farejador de pacotes normalmente permitiria. É bom saber que essas opções existem.

Sobre o comando ip

O comando `ip` é um utilitário bastante recente, usado para visualizar e definir quase tudo relacionado às configurações de rede. Esta lição cobre alguns dos subcomandos mais usados do `ip`, embora mal arranhe a superfície do que está disponível. Vale a pena ler a documentação para aprender a usá-lo da maneira mais eficaz.

Cada subcomando de `ip` tem sua própria página de manual. A seção `SEE ALSO` da página do manual de `ip` inclui uma lista deles:

```
$ man ip
...
SEE ALSO
    ip-address(8), ip-addrlabel(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
...
```

Em vez de passar por aqui toda vez que você precisar da página de manual, simplesmente adicione `-e` o nome do subcomando a `ip`; por exemplo, `man ip-route`.

Outra boa fonte de informações é a função de ajuda. Para visualizar a ajuda integrada, adicione `help` após o subcomando:

```
$ 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
...
```

Máscara de rede e revisão de roteamento

O IPv4 e o IPv6 são conhecidos como protocolos roteados ou roteáveis. Isso significa que eles são projetados de forma a permitir que os designers de rede controlem o fluxo de tráfego. O Ethernet não é um protocolo roteável. Assim, se você conectar diversos dispositivos somente com Ethernet, haverá pouquíssimas opções para controlar o fluxo de tráfego da rede. Quaisquer medidas empregadas para controlar o tráfego acabariam tendo resultado semelhante aos protocolos roteáveis e de roteamento atuais.

Os protocolos roteáveis permitem que os designers de rede segmentem as redes para reduzir os requisitos de processamento dos dispositivos de conectividade, fornecer redundância e gerenciar o tráfego.

Os endereços IPv4 e IPv6 têm duas seções. O primeiro conjunto de bits constitui a seção da rede, ao passo que o segundo representa a seção do host. O número de bits que constituem a seção da rede é determinado pela *máscara de rede* (também chamada de *máscara de sub-rede*). Ele também pode ser chamado de *comprimento do prefixo*. Independentemente de como é chamado, trata-se do número de bits que a máquina trata como sendo a parte da rede do endereço. No IPv4, ele às vezes é especificado em notação decimal com pontos.

Veja abaixo um exemplo usando IPv4. Observe como os dígitos binários mantêm seu valor de posição nos octetos, mesmo quando são divididos pela máscara de rede.

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

A seção da rede de um endereço é usada pelas máquinas IPv4 ou IPv6 para pesquisar em qual interface um pacote deve ser enviado em sua tabela de roteamento. Quando um host IPv4 ou IPv6 com roteamento habilitado recebe um pacote que não é destinado ao próprio host, ele busca uma correspondência para a parte da rede do destino em uma rede da tabela de roteamento. Se uma entrada correspondente for encontrada, ele envia o pacote ao destino especificado na tabela. Se nenhuma entrada for encontrada e uma rota padrão estiver configurada, ele é enviado para a rota padrão. Se nenhuma entrada for encontrada e nenhuma rota padrão estiver configurada, o pacote é descartado.

Configurando uma interface

Vamos falar de duas ferramentas que permitem configurar uma interface de rede: `ifconfig` e `ip`. O programa `ifconfig`, embora ainda amplamente utilizado, é considerado uma ferramenta legada e nem sempre está disponível em sistemas mais novos.

TIP Nas distribuições Linux mais recentes, a instalação do pacote `net-tools` fornece os comandos de rede legados.

Antes de configurar uma interface, devemos primeiro saber quais interfaces estão disponíveis. Há algumas maneiras de fazer isso. Uma delas é usar a opção `-a` de `ifconfig`:

```
$ ifconfig -a
```

Outra forma é usar `ip`. Há diversos exemplos com `ip addr`, `ip a` e alguns com `ip address`. Eles são todos sinônimos. Oficialmente, o subcomando é `ip address`. Assim, para visualizar a página de manual, usamos `man ip-address` e não `man ip-addr`.

O subcomando `link` para `ip` lista os links de interface disponíveis para configuração:

```
$ 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
```

Pressupondo que o sistema de arquivos `sys` esteja montado, também podemos listar o conteúdo de `/sys/class/net`

```
$ ls /sys/class/net
enp0s3  enp0s8  lo
```

Para configurar uma interface com o `ifconfig`, você deve estar logado como root ou rodar um utilitário como `sudo` para executar o comando com privilégios de root. Siga o exemplo abaixo:

```
# ifconfig enp1s0 192.168.50.50/24
```

A versão Linux do ifconfig é flexível com a forma de se especificar a máscara de sub-rede:

```
# ifconfig eth2 192.168.50.50 netmask 255.255.255.0
# ifconfig eth2 192.168.50.50 netmask 0xffffffff00
# ifconfig enp0s8 add 2001:db8::10/64
```

Note como, no IPv6, a palavra-chave `add` foi usada. Se um endereço IPv6 não for precedido por `add`, será exibida uma mensagem de erro.

O seguinte comando configura uma interface com ip:

```
# ip addr add 192.168.5.5/24 dev enp0s8
# ip addr add 2001:db8::10/64 dev enp0s8
```

No caso do ip, o mesmo comando é usado para IPv4 e IPv6.

Configurando opções de baixo nível

O comando `ip link` é usado para configurar a interface de baixo nível ou para configurações de protocolos, como VLANs, ARP ou MTUs, ou ainda para desabilitar uma interface.

Uma tarefa comum para `ip link` é desabilitar ou habilitar uma interface. Isso também pode ser feito com o 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
```

Às vezes é necessário ajustar o MTU de uma interface. Da mesma forma que é possível habilitar/desabilitar interfaces, esse ajuste também pode ser feito com ifconfig ou 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
```

A tabela de roteamento

Os comandos `route`, `netstat -r` e `ip route` podem ser usados para visualizar a tabela de roteamento. Se quiser modificar as rotas, use `route` ou `ip route`. Eis abaixo alguns exemplos de visualização de uma tabela de roteamento:

```
$ 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
```

Observe como não há saída relacionada ao IPv6. Para visualizar a tabela de roteamento do IPv6, usamos `route -6`, `netstat -6r` e `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

Um exemplo de `netstat -r6` foi omitido porque sua saída é idêntica à de `route -6`. Algumas das saídas do comando `route` acima são autoexplicativas. A coluna `Flag` fornece algumas informações sobre a rota. O sinalizador `U` indica que uma rota está ativa. Um `!` significa rota rejeitada, ou seja, uma rota com um `!` não será usada. O sinalizador `n` indica que a rota não foi armazenada em cache. O kernel mantém um cache de rotas separadamente de todas as rotas conhecidas para permitir pesquisas mais rápidas. O sinalizador `G` indica um gateway. A coluna `Metric` ou `Met` não é usada pelo kernel. Ela se refere à distância administrativa até o alvo. Essa distância administrativa é usada pelos protocolos de roteamento para determinar rotas dinâmicas. A coluna `Ref` é a contagem de referência ou o número de usos de uma rota. Assim como `Metric`, ela não é usada pelo kernel do Linux. A coluna `Use` mostra o número de pesquisas para uma rota.

Na saída de `netstat -r`, `MSS` indica o tamanho máximo do segmento para as conexões TCP nessa rota. A coluna `Window` mostra o tamanho da janela TCP padrão. `irtt` mostra o tempo de ida e volta dos pacotes nesta rota.

A saída de `ip route` e `ip -6 route` é interpretada desta maneira:

1. Destino.
2. Endereço opcional seguido pela interface.
3. O protocolo de roteamento usado para adicionar a rota.
4. O escopo da rota. Se omitido, trata-se de um escopo global ou um gateway.
5. A métrica da rota. Ela é usada pelos protocolos de roteamento dinâmico para determinar o custo da rota. Não é usada pela maioria dos sistemas.
6. Se for uma rota IPv6, a preferência da rota RFC4191.

Para que isso fique mais claro, eis alguns exemplos:

Exemplo de IPv4

```
default via 10.0.2.2 dev enp0s3 proto dhcp metric 100
```

1. O destino é a rota padrão.
2. O endereço do gateway é 10.0.2.2, acessível através da interface enp0s3.
3. Ela foi adicionada à tabela de roteamento pelo DHCP.
4. O escopo foi omitido, portanto é global.
5. A rota tem um valor de custo de 100.
6. Não há preferência de rota de IPv6.

Exemplo de IPv6

```
fc0::/64 dev enp0s8 proto kernel metric 256 pref medium
```

1. O destino é fc0::/64.
2. Ele é acessível através da interface enp0s8.
3. Foi adicionado automaticamente pelo kernel.
4. O escopo foi omitido, portanto é global.
5. A rota tem um valor de custo de 256.
6. Ela tem uma preferência de IPv6 medium.

Gerenciando as rotas

As rotas podem ser gerenciadas com `route` ou `ip route`. Veja abaixo um exemplo de adição e remoção de uma rota usando o comando `route`. Com `route`, adicionamos a opção `-6` para 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
```

Abaixo, o mesmo exemplo usando o comando `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
```

Exercícios Guiados

1. Quais comandos podem ser usados para listar as interfaces de rede?

2. Como desativar temporariamente uma interface? E como reativá-la?

3. Qual das opções a seguir é uma máscara de sub-rede razoável para IPv4?

0.0.0.255	
255.0.255.0	
255.252.0.0	
/24	

4. Quais comandos você pode usar para verificar sua rota padrão?

5. Como adicionar um segundo endereço IP a uma interface?

Exercícios Exploratórios

1. Qual subcomando de `ip` pode ser usado para configurar a marcação de `vlan`?

2. Como configurar uma rota padrão?

3. Como obter informações detalhadas sobre o comando `ip neighbour`? O que acontece se ele for executado sozinho?

4. Como fazer backup da tabela de roteamento? Como restaurar esse backup?

5. Qual subcomando de `ip` pode ser usado para configurar opções de spanning tree?

Resumo

Em geral, as redes são configuradas pelos scripts de inicialização do sistema ou por um ajudante como o NetworkManager. A maioria das distribuições possui ferramentas para editar os arquivos de configuração do script de inicialização. Consulte a documentação de sua distribuição para saber mais.

A possibilidade de configurar manualmente a rede permite solucionar problemas com mais eficácia. Isso é muito útil nos ambientes mínimos usados para tarefas como a restauração de backups ou a migração para um novo hardware.

Os utilitários abordados nesta seção têm mais funcionalidades do que as mostradas. Vale a pena estudar a página de manual de cada um deles para se familiarizar com as opções disponíveis. Os comandos `ss` e `ip` são a maneira moderna de fazer as coisas, enquanto os outros recursos, embora ainda presentes, são considerados ferramentas legadas.

A melhor maneira de se familiarizar com as ferramentas abordadas é praticar. Usando um computador com uma quantidade modesta de RAM, é possível configurar um laboratório de rede virtual usando máquinas virtuais para suas experiências. Três máquinas virtuais já bastam para você se familiarizar com as ferramentas listadas.

Os comandos usados nesta lição incluem:

ifconfig

Utilitário legado usado para configurar interfaces de rede e revisar seus estados.

ip

Utilitário moderno e versátil usado para configurar interfaces de rede e revisar seus estados.

netstat

Comando legado usado para visualizar as conexões de rede atuais e as informações de rota.

route

Comando legado usado para visualizar ou modificar a tabela de roteamento de um sistema.

Respostas aos Exercícios Guiados

1. Quais comandos podem ser usados para listar as interfaces de rede?

Qualquer um dos comandos abaixo:

```
ip link, ifconfig -a, ou ls /sys/class/net
```

2. Como desativar temporariamente uma interface? E como reativá-la?

Podemos usar `ifconfig` ou `ip link`:

Usando `ifconfig`:

```
$ ifconfig wlan1 down
$ ifconfig wlan1 up
```

Usando `ip link`:

```
$ ip link set wlan1 down
$ ip link set wlan1 up
```

3. Qual das opções a seguir é uma máscara de sub-rede razoável para IPv4?

- 255.252.0.0
- /24

As outras máscaras listadas são inválidas porque não separam o endereço de forma clara em duas seções, sendo que a primeira parte define a rede e a segunda o host. Os bits mais à esquerda de uma máscara sempre serão 1 e os bits à direita sempre serão 0.

4. Quais comandos você pode usar para verificar sua rota padrão?

Podemos usar `route`, `netstat -r`, ou `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. Como adicionar um segundo endereço IP a uma interface?

Usaríamos `ip address` ou `ifconfig`. Lembre-se de que `ifconfig` é uma ferramenta legada:

```
$ ip addr add 172.16.15.16/16 dev enp0s9 label enp0s9:sub1
```

A parte `label enp0s9:sub1` do comando adiciona um alias a `enp0s9`. Se você não usa o comando legado `ifconfig`, pode omitir essa parte. Caso contrário, o comando ainda vai funcionar, mas o endereço adicionado não aparecerá na saída de `ifconfig`.

Também é possível usar o `ifconfig`:

```
$ ifconfig enp0s9:sub1 172.16.15.16/16
```

Respostas aos Exercícios Exploratórios

1. Qual subcomando de ip pode ser usado para configurar a marcação de vlan?

ip link tem uma opção `vlan` que pode ser empregada. Veja abaixo um exemplo de marcação de uma sub-interface com `vlan 20`.

```
# ip link add link enp0s9 name enp0s9.20 type vlan id 20
```

2. Como configurar uma rota padrão?

Usando `route` ou `ip route`:

```
# route add default gw 192.168.1.1
# ip route add default via 192.168.1.1
```

3. Como obter informações detalhadas sobre o comando `ip neighbour`? O que acontece se ele for executado sozinho?

Lendo a página de manual:

```
$ man ip-neighbour
```

Ele exibe o cache ARP:

```
$ ip neighbour
10.0.2.2 dev enp0s3 lladdr 52:54:00:12:35:02 REACHABLE
```

4. Como fazer backup da tabela de roteamento? Como restaurar esse backup?

O exemplo abaixo demonstra o backup e a restauração de uma tabela de roteamento:

```
# ip route save > /root/routes/route_backup
# ip route restore < /root/routes/route_backup
```

5. Qual subcomando de ip pode ser usado para configurar opções de spanning tree?

Como no caso do gerenciamento de configurações de vlan, o `ip link` pode configurar o spanning tree usando o tipo `bridge`. O exemplo mostra a adição de uma interface virtual com

uma prioridade STP de 50:

```
# ip link add link enp0s9 name enp0s9.50 type bridge priority 50
```



109.3 Lição 2

Certificação:	LPIC-1
Versão:	5.0
Tópico:	109 Fundamentos de rede
Objetivo:	109.3 Resolução de problemas básicos de rede
Lição:	2 de 2

Introdução

Os sistemas operacionais baseados em Linux incluem uma variedade de ferramentas para solucionar problemas de rede. Esta lição abordará algumas das mais comuns. Neste ponto, você deve ter uma noção do modelo OSI ou de outros modelos de rede em camadas, endereçamento IPv4 ou IPv6 e os fundamentos de roteamento e comutação.

A melhor maneira de testar uma conexão de rede é tentar usar seu aplicativo. Se isso não funcionar, existem muitas ferramentas disponíveis para ajudar a diagnosticar o problema.

Testando conexões com ping

Os comandos `ping` e `ping6` enviam uma solicitação de eco ICMP para um endereço IPv4 ou IPv6, respectivamente. Uma solicitação de eco ICMP envia uma pequena quantidade de dados ao endereço de destino. Se o endereço de destino estiver acessível, ele manda uma mensagem de eco ICMP de volta ao remetente com os mesmos dados que lhe foram enviados:

```
$ 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
```

A opção `-c` é usada para especificar o número de pacotes a enviar. Se você omitir esta opção, `ping` e `ping6` continuarão a enviar pacotes até que você os interrompa, normalmente com a combinação de teclas `ctrl + c`

Não é possível executar `ping` em um host, mas isso não significa que você não pode se conectar a ele. Muitas empresas têm firewalls ou listas de controle de acesso ao roteador que bloqueiam tudo, exceto o mínimo necessário para que os sistemas funcionem. Isso inclui as solicitações e respostas de eco ICMP. Como esses pacotes podem incluir dados arbitrários, um invasor astuto poderia usá-los para extrair dados.

Traçando Rotas

Os programas `traceroute` e `traceroute6` servem para mostrar a rota que um pacote faz para chegar ao seu destino. Para isso, eles enviam diversos pacotes ao destino, incrementando o campo *Time-To-Live* (TTL) do cabeçalho IP com cada pacote subsequente. Cada roteador ao longo do caminho responde com uma mensagem ICMP de TTL excedido:

```
$ 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

```

Por padrão, o `traceroute` envia 3 pacotes UDP com dados inúteis para a porta 33434, incrementando-os a cada vez que envia um pacote. Cada linha na saída do comando é uma interface de roteador que o pacote atravessa. Os tempos mostrados em cada linha da saída são o tempo de ida e volta de cada pacote. O endereço IP é o endereço da interface do roteador em questão. Se o `traceroute` puder, ele usa o nome DNS da interface do roteador. Às vezes, aparece * no lugar de uma hora. Quando isso acontece, significa que o `traceroute` nunca recebeu a mensagem de TTL excedido para este pacote. Isso geralmente indica que a última resposta é o último salto da rota.

Se você tiver acesso a root, a opção `-I` configura o `traceroute` para usar solicitações de eco ICMP ao invés de pacotes UDP. Elas geralmente são mais eficazes do que o UDP porque o host de destino tem mais probabilidade de responder a uma solicitação de eco ICMP do que ao pacote 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 dtr01hlrgnc-gbe-4-15.hlrg.nc.charter.com (96.34.64.172) 8.763 ms 8.775 ms 8.770 ms
4 acr01mgtnnc-vln-492.mgtn.nc.charter.com (96.34.67.202) 27.080 ms 27.154 ms 27.151 ms
5 bbr01gnvlsc-bue-3.gnvl.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
```

Algumas empresas bloqueiam solicitações e respostas de eco ICMP. Para contornar isso, podemos usar o TCP. Usando uma porta TCP aberta conhecida, garantimos que o host de destino responderá. Para usar o TCP, inclua a opção **-T** junto com **-p** para especificar a porta. Como no caso das solicitações de eco ICMP, é necessário ter acesso de root para fazer isso:

```
# 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 dtr01hlrgnc-gbe-4-15.hlrg.nc.charter.com (96.34.64.172) 12.134 ms 12.093 ms 12.062 ms
4 acr01mgtnnc-vln-492.mgtn.nc.charter.com (96.34.67.202) 31.146 ms 31.192 ms 31.828 ms
5 bbr01gnv1sc-bue-3.gnvl.sc.charter.com (96.34.2.112) 39.057 ms 46.706 ms 39.745 ms
6 bbr01alndlmi-tge-0-0-0-13.alndl.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
```

Como o ping, o traceroute tem suas limitações. É possível que firewalls e roteadores bloqueiem os pacotes enviados ou devolvidos ao traceroute. Se você tiver acesso de root, há opções que podem ajudar a obter resultados precisos.

Encontrando MTUs com tracepath

O comando `tracepath` é semelhante ao `traceroute`. A diferença é que ele rastreia os tamanhos da *Maximum Transmission Unit* (MTU) ao longo do caminho. A MTU é uma configuração definida em uma interface de rede, ou uma limitação de hardware sobre a maior unidade de dados de protocolo que é possível transmitir ou receber. O programa `tracepath` funciona da mesma maneira que `traceroute` no sentido de que incrementa o TTL a cada pacote. A diferença é o envio de um datagrama UDP muito grande. É quase inevitável que o datagrama seja maior do que o dispositivo com a menor MTU ao longo da rota. Quando o pacote chega a este dispositivo, este normalmente responde com um pacote de destino inacessível. O pacote de destino ICMP inacessível tem um campo para a MTU do link para o qual ele enviaria o pacote se pudesse. O `tracepath` então manda todos os pacotes subsequentes com esse tamanho:

```
$ 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
```

Ao contrário do `traceroute`, precisamos obrigatoriamente usar o `tracepath6` para 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
```

A saída é semelhante à do `traceroute`. A vantagem do `tracepath` está na última linha, que inclui o menor MTU de todo o link. Isso pode ser útil para solucionar problemas de conexões incapazes de lidar com fragmentos.

Como no caso das ferramentas anteriores de resolução de problemas, existe a possibilidade de o equipamento bloquear seus pacotes.

Criando Conexões Arbitrárias

O programa `nc`, conhecido como `netcat`, pode enviar ou receber dados arbitrários através de uma

conexão de rede TCP ou UDP. Os exemplos a seguir devem deixar clara sua funcionalidade.

Aqui está um exemplo de configuração de um ouvinte (listener) na porta 1234:

```
$ nc -l 1234
LPI Example
```

A saída de LPI Example aparece após o exemplo abaixo, que mostra a configuração de um remetente netcat para enviar pacotes para net2.example.net na porta 1234. A opção `-l` é usada para especificar que nc deve receber dados em vez de enviá-los:

```
$ nc net2.example.net 1234
LPI Example
```

Pressione `Ctrl + C` em qualquer um dos sistemas para interromper a conexão.

O netcat funciona com endereços IPv4 e IPv6. Ele funciona com TCP e UDP. Pode inclusive ser usado para configurar um shell remoto rudimentar.

WARNING Nem toda instalação de nc suporta a opção `-e`. Consulte as páginas de manual de sua instalação para obter informações de segurança sobre esta opção, bem como métodos alternativos para executar comandos em um sistema remoto.

```
$ hostname
net2
$ nc -u -e /bin/bash -l 1234
```

A opção `-u` significa UDP. `-e` instrui o netcat a enviar tudo o que recebe para a entrada padrão do executável que vem em seguida. Neste exemplo, `/bin/bash`.

```
$ hostname
net1
$ nc -u net2.example.net 1234
hostname
net2
pwd
/home/emma
```

Percebeu como a saída do comando `hostname` corresponde à do host ouvinte e a saída do comando `pwd` é um diretório?

Visualizando conexões atuais e listeners

Os programas `netstat` e `ss` podem ser usados para visualizar o status de seus ouvintes e conexões atuais. Como no caso de `ifconfig`, `netstat` é uma ferramenta legada. Tanto `netstat` quanto `ss` têm saídas e opções semelhantes. Eis algumas opções disponíveis para ambos os programas:

-a

Mostra todos os sockets.

-l

Mostra os sockets de escuta.

-p

Mostra o processo associado à conexão.

-n

Impede pesquisas de nome para portas e endereços.

-t

Mostra as conexões TCP.

-u

Mostra as conexões UDP.

Os exemplos abaixo mostram a saída de um conjunto de opções comumente usado em ambos os programas:

```
# 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:*              LISTEN    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                      :22                  *:
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))
```

A coluna `Recv-Q` é o número de pacotes que um socket recebeu, mas não passou para seu programa. A coluna `Send-Q` é o número de pacotes que um socket enviou e que não foram confirmados pelo receptor. As colunas restantes são autoexplicativas.

Exercícios Guiados

1. Que comando(s) usariamos para enviar um eco ICMP para learning.lpi.org?

2. Como seria possível determinar a rota até 8.8.8.8?

3. Qual comando poderia mostrar se há processos escutando na porta TCP 80?

4. Como descobrir qual processo está escutando em uma porta?

5. Como determinar a MTU máxima de um caminho de rede?

Exercícios Exploratórios

1. Como poderíamos usar o netcat para enviar uma solicitação HTTP a um servidor web?

2. Por quais motivos o ping em um host pode falhar?

3. Cite uma ferramenta que poderia ser usada para ver os pacotes de rede que estão chegando ou saindo de um host Linux.

4. Como forçar o traceroute a usar uma interface diferente?

5. O traceroute é capaz de relatar MTUs?

Resumo

Em geral, as redes são configuradas pelos scripts de inicialização do sistema ou por um ajudante como o NetworkManager. A maioria das distribuições possui ferramentas para editar os arquivos de configuração do script de inicialização. Consulte a documentação de sua distribuição para saber mais.

A possibilidade de configurar manualmente a rede permite solucionar problemas com mais eficácia. Isso é muito útil nos ambientes mínimos usados para tarefas como a restauração de backups ou a migração para um novo hardware.

Os utilitários abordados nesta seção têm mais funcionalidades do que as mostradas. Vale a pena estudar a página de manual de cada um deles para se familiarizar com as opções disponíveis. Os comandos `ss` e `ip` são a maneira moderna de fazer as coisas, enquanto os outros recursos, embora ainda presentes, são considerados ferramentas legadas.

A melhor maneira de se familiarizar com as ferramentas abordadas é praticar. Usando um computador com uma quantidade modesta de RAM, é possível configurar um laboratório de rede virtual usando máquinas virtuais para suas experiências. Três máquinas virtuais já bastam para você se familiarizar com as ferramentas listadas.

Os comandos discutidos nesta lição foram:

ping e ping6

Usados para transmitir pacotes ICMP para um host remoto a fim de testar a disponibilidade de uma conexão de rede.

traceroute e traceroute6

Usados para rastrear um caminho através de uma rede para determinar a conectividade da rede.

tracepath e tracepath6

Usados para rastrear um caminho através de uma rede, bem como determinar os tamanhos de MTU ao longo de uma rota.

nc

Usado para configurar conexões arbitrárias em uma rede para testar a conectividade, bem como consultar uma rede sobre os serviços e dispositivos disponíveis.

netstat

Comando legado usado para determinar as estatísticas e conexões de rede abertas de um

sistema.

ss

Comando moderno usado para determinar as estatísticas e conexões de rede abertas de um sistema.

Respostas aos Exercícios Guiados

1. Que comando(s) usariamos para enviar um eco ICMP para learning.lpi.org?

Usaríamos ping ou ping6:

```
$ ping learning.lpi.org
```

ou

```
$ ping6 learning.lpi.org
```

2. Como seria possível determinar a rota até 8.8.8.8?

Usando os comandos tracepath ou traceroute.

```
$ tracepath 8.8.8.8
```

ou

```
$ traceroute 8.8.8.8
```

3. Qual comando poderia mostrar se há processos escutando na porta TCP 80?

Com ss:

```
$ ss -ln | grep ":80"
```

Com netstat:

```
$ netstat -ln | grep ":80"
```

Embora este não seja um requisito para o exame, você também poderia usar lsof:

```
# lsof -Pi:80
```

4. Como descobrir qual processo está escutando em uma porta?

Mais uma vez, existem diversas maneiras de se fazer isso. Podemos usar o `lsof` como na resposta anterior, substituindo o número da porta. Também podemos usar `netstat` ou `ss` com a opção `-p`. Lembre-se de que o `netstat` é considerado uma ferramenta legada.

```
# netstat -lnp | grep ":22"
```

As mesmas opções que funcionam com `netstat` também funcionam com `ss`:

```
# ss -lnp | grep ":22"
```

5. Como determinar a MTU máxima de um caminho de rede?

Usando o comando `tracepath`:

```
$ tracepath somehost.example.com
```

Respostas aos Exercícios Exploratórios

1. Como poderíamos usar o netcat para enviar uma solicitação HTTP a um servidor web?

Inserindo a linha de solicitação HTTP, os cabeçalhos necessários e uma linha em branco no terminal:

```
$ 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. Por quais motivos o ping em um host pode falhar?

Há uma série de possíveis razões. Eis algumas delas:

- O host remoto está inativo.
- Um roteador ACL está bloqueando seu ping.
- O firewall do host remoto está bloqueando seu ping.
- Você pode estar usando um nome de host ou endereço incorreto.
- A resolução do seu nome está retornando um endereço incorreto.
- A configuração de rede da sua máquina está incorreta.
- O firewall da sua máquina está bloqueando o ping.
- A configuração de rede do host remoto está incorreta.
- As interfaces de sua máquina estão desconectadas.
- As interfaces da máquina remota estão desconectadas.
- Um componente de rede como um switch, cabo ou roteador entre sua máquina e a máquina remota não está mais funcionando.

3. Cite uma ferramenta que poderia ser usada para ver os pacotes de rede que estão chegando ou saindo de um host Linux.

Tanto `tcpdump` quanto `wireshark` fariam o serviço.

4. Como forçar o `traceroute` a usar uma interface diferente?

Usando a opção `-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. O `traceroute` é capaz de relatar MTUs?

Sim, com a opção `--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 Configurar DNS cliente

Referência ao LPI objectivo

LPIC-1 version 5.0, Exam 102, Objective 109.4

Peso

2

Áreas chave de conhecimento

- Consultar servidores DNS remotos.
- Configurar a resolução local de nomes e o uso de servidores DNS remotos.
- Modificar a ordem em que a resolução de nomes é feita.
- Identificar erros relacionados à resolução de nomes.
- Noções do `systemd-resolved`.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- `/etc/hosts`
- `/etc/resolv.conf`
- `/etc/nsswitch.conf`
- `host`
- `dig`
- `getent`



**Linux
Professional
Institute**

109.4 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	109 Fundamentos de rede
Objetivo:	109.4 Configurar o DNS do lado do cliente
Lição:	1 de 1

Introdução

Esta lição trata da configuração da resolução de nomes do lado do cliente. Falaremos também do uso de algumas ferramentas da linha de comando para resolução de nomes.

Não é viável memorizar e manter endereços IP, UIDs e GIDs e tantos outros números. Os serviços de resolução de nomes traduzem nomes fáceis de memorizar em números e vice-versa. Esta lição enfoca a resolução do nome do host, mas um processo semelhante ocorre para nomes de usuário, nomes de grupo, números de porta e muitos outros.

O processo de resolução de nome

Os programas que resolvem nomes em números quase sempre usam funções fornecidas pela biblioteca C padrão, que nos sistemas Linux é a glibc do projeto GNU. A primeira coisa que essas funções fazem é ler o arquivo `/etc/nsswitch.conf` para obter instruções sobre como resolver esse tipo de nome. Esta lição trata da resolução de nomes de host, mas o mesmo processo também se aplica a outros tipos de resolução de nomes. Depois de ler `/etc/nsswitch.conf`, o processo busca pelo nome da maneira especificada. Como `/etc/nsswitch.conf` suporta plug-ins, o que vem a seguir pode ser qualquer coisa. Depois que a função termina de buscar o nome ou número,

ela retorna o resultado para o processo de chamada.

Classes DNS

O DNS tem três classes de registro, IN, HS e CH. Nesta lição, todas as consultas de DNS serão do tipo IN. A classe IN se refere a endereços da internet que usam a pilha TCP/IP. CH significa ChaosNet, uma tecnologia de rede que teve vida curta e não está mais em uso. A classe HS refere-se ao Hesiod. O Hesiod é uma maneira de armazenar coisas como passwd e entradas de grupo no DNS. O Hesiod está além do escopo desta lição.

Entendendo /etc/nsswitch.conf

A melhor maneira de aprender mais sobre este arquivo é ler a página de manual que faz parte do projeto de páginas de manual do Linux. Ela está disponível na maioria dos sistemas. Pode ser acessada com o comando `man nsswitch.conf`. Como alternativa, ela pode ser encontrada em https://man7.org/linux/man-pages/dir_section_5.html

Veja abaixo um exemplo simples de `/etc/nsswitch.conf` tirado da man page:

```
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.
```

O arquivo é organizado em colunas. A coluna mais à esquerda é o tipo de banco de dados de nomes. O resto das colunas são os métodos que as funções de resolução devem usar para pesquisar um nome. Os métodos são seguidos pelas funções, da esquerda para a direita. As colunas com `[]` são usadas para fornecer uma lógica condicional limitada para a coluna imediatamente à esquerda.

Suponha que um processo esteja tentando resolver o nome do host `learning.lpi.org`. Ele faria uma chamada apropriada à biblioteca C (provavelmente `gethostbyname`). Esta função então lê `/etc/nsswitch.conf`. Como o processo está procurando um nome de host, ele encontrará a linha que começa com `hosts`. Em seguida, ele tenta usar o DNS para resolver o nome. A coluna

seguinte, [!UNAVAIL=return] indica que, se o serviço *não* estiver indisponível, não é necessário tentar a próxima fonte; ou seja, se o DNS estiver disponível, ele para de tentar resolver o nome do host, mesmo se os servidores de nome forem incapazes de fazê-lo. Se o DNS estiver indisponível, ele prossegue para a próxima fonte. Neste caso, a próxima fonte é files.

Quando vemos uma coluna no formato [resultado=ação], isso significa que quando uma pesquisa do resolvedor na coluna à esquerda resultar em resultado, então a ação é executada. Se resultado for precedido por !, isso quer dizer que, se o resultado *não* for resultado, a ação deve ser executada. Para obter descrições dos resultados e ações possíveis, consulte a página do manual.

Agora, suponha que um processo esteja tentando resolver um número de porta para um nome de serviço. Ele lerá a linha services. A primeira fonte listada é NIS. NIS significa *Network Information Service* (às vezes apelidado de Páginas Amarelas). Este é um serviço antigo que permitia o gerenciamento central de coisas como usuários. Raramente é usado devido à sua fraca segurança. A coluna seguinte, [NOTFOUND=return], indica que se a pesquisa foi bem-sucedida, mas o serviço não foi encontrado, é preciso parar de procurar. Se a condição acima mencionada não se aplicar, ele usará arquivos locais.

Qualquer coisa à direita de # é um comentário e é ignorado.

O arquivo /etc/resolv.conf

O arquivo /etc/resolv.conf é usado para configurar a resolução do host via DNS. Algumas distribuições têm scripts de inicialização, daemons e outras ferramentas que gravam neste arquivo. Lembre-se disso ao editá-lo manualmente. Verifique a documentação de sua distribuição e de quaisquer ferramentas de configuração de rede, se esse for o seu caso. Algumas ferramentas, como o NetworkManager, deixam um comentário no arquivo informando que as alterações manuais serão sobreescritas.

Como no caso de /etc/nsswitch.conf, existe uma página de manual associada ao arquivo. Ela pode ser acessada com o comando `man resolv.conf` ou em <https://man7.org/linux/man-pages/man5/resolv.conf.5.html>.

O formato do arquivo é bastante simples. Na coluna da esquerda, temos a opção name. O resto das colunas na mesma linha são os valores da opção.

A opção mais comum é nameserver. Ela é usada para especificar o endereço IPv4 ou IPv6 de um servidor DNS. Na data em que esta lição foi escrita, era possível especificar até três servidores de nomes. Se o seu /etc/resolv.conf não tiver a opção nameserver, o sistema usará por padrão o servidor de nomes da máquina local.

Veja abaixo um exemplo simples, mas representativo de configurações que são comuns:

```
search lpi.org
nameserver 10.0.0.53
nameserver fd00:ffff::2:53
```

A opção `search` é usada para permitir pesquisas curtas. No exemplo, configuramos um único domínio para pesquisa, `lpi.org`. Isso significa que qualquer tentativa de resolver um nome de host sem uma parte de domínio terá `.lpi.org` incluído antes da pesquisa. Por exemplo, se pesquisarmos por um host chamado `learning`, o resolvedor buscará por `learning.lpi.org`. É possível configurar até seis domínios de pesquisa.

Outra opção comum é `domain`. Ela é usada para definir o nome de domínio local. Se esta opção estiver ausente, o padrão é usar tudo após o primeiro `.` no nome de host da máquina. Se o nome do host não contiver um `.`, presume-se que a máquina faça parte do domínio raiz. Como no caso de `search`, `domain` pode ser usado para pesquisas curtas de nomes.

Lembre-se de que `domain` e `search` são mutuamente exclusivos. Se ambos estiverem presentes, será usada a última instância no arquivo.

Muitas opções podem ser definidas de forma a afetar o comportamento do resolvedor. Para configurá-las, use a palavra-chave `option`, seguida pelo nome da opção a ser configurada e, se for o caso, um `:` seguido pelo valor. Veja abaixo um exemplo de configuração da opção de tempo limite (`timeout`), ou seja, o período de tempo em segundos que o resolvedor espera por um servidor de nome antes de desistir:

```
option timeout:3
```

Existem outras opções para `resolv.conf`, mas essas são as mais comuns.

O arquivo `/etc/hosts`

O arquivo `/etc/hosts` é usado para resolver nomes para endereços IP e vice-versa. Há suporte a IPv4 e IPv6. A coluna da esquerda é o endereço IP, o resto são nomes associados a esse endereço. O uso mais comum de `/etc/hosts` é para hosts e endereços nos quais o DNS não é possível, como endereços de loopback. No exemplo abaixo, são definidos os endereços IP dos componentes críticos da infraestrutura.

Eis um exemplo realístico de um arquivo `/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

O systemd inclui um serviço chamado `systemd-resolved`. Ele fornece mDNS, DNS e LLMNR. Quando está em execução, ele escuta as solicitações de DNS em 127.0.0.53. Ele *não* fornece um servidor DNS completo. Quaisquer solicitações de DNS que recebe são pesquisadas nos servidores configurados em `/etc/systemd/resolv.conf` ou `/etc/resolv.conf`. Se você deseja empregar o serviço, use `resolve` para hosts em `/etc/nsswitch.conf`. Lembre-se de que o pacote do sistema operacional que possui a biblioteca `systemd-resolution` pode não estar instalado por padrão.

Ferramentas de resolução de nomes

Existem muitas ferramentas de resolução de nomes disponíveis para os usuários do Linux. Esta lição cobre três delas. A primeira, `getent`, é útil para ver como as solicitações do mundo real serão resolvidas. Outro comando é `host`, que é ótimo para consultas de DNS simples. Um programa chamado `dig` é prático para operações de DNS complexas que podem ajudar na resolução de problemas do servidor DNS.

O comando `getent`

O utilitário `getent` é usado para exibir entradas de bancos de dados de serviços de nome. Ele é capaz de recuperar registros de qualquer fonte configurável por `/etc/nsswitch.conf`.

Para usar o `getent`, inclua o tipo de nome que deseja resolver após o comando e, opcionalmente, uma entrada específica a pesquisar. Se você especificar apenas o tipo de nome, o `getent` tentará exibir todas as entradas referentes àquele tipo de dados:

```
$ 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
```

A partir da versão 2.2.5 do glibc, é possível forçar o `getent` a usar uma fonte de dados específica com a opção `-s`. O exemplo abaixo demonstra isso:

```
$ 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
```

O comando host

`host` é um programa simples para procurar entradas DNS. Sem opções, se `host` receber um nome, ele retorna os conjuntos de registros A, AAAA e MX. Se um endereço IPv4 ou IPv6 for fornecido, ele produzirá o registro PTR, caso haja um disponível:

```
$ 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.
```

Se você estiver procurando por um tipo de registro específico, pode usar `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 também pode ser usado para consultar um servidor de nomes específico caso não se queira usar os que estão em `/etc/resolv.conf`. Basta adicionar o endereço IP ou nome de host do servidor que se deseja usar como último argumento:

```
$ 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.
```

O comando dig

Outra ferramenta para consultar servidores DNS é o dig. Esse comando é muito mais detalhado do que host. Por padrão, dig consulta os registros A. Ele provavelmente é prolixo demais para simplesmente buscar um endereço IP ou nome de host. O dig funciona para pesquisas simples, mas é mais adequado para solucionar problemas de configuração do servidor 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

```

Como vemos, o `dig` fornece muitas informações. A saída é dividida em seções. A primeira seção exibe informações sobre a versão do `dig` instalada e a consulta enviada, junto com as opções usadas para o comando. Em seguida vêm informações sobre a consulta e a resposta.

A seção seguinte mostra informações sobre as extensões EDNS usadas e a consulta. No exemplo, a extensão cookie é usada. O `dig` está procurando um registro A para `learning.lpi.org`.

A seção seguinte mostra o resultado da consulta. O número na segunda coluna é o TTL do recurso em segundos.

O restante da saída fornece informações sobre os servidores de nome do domínio, incluindo os registros NS para o servidor, junto com os registros A e AAAA dos servidores no registro NS do domínio.

Como `host`, você pode especificar um tipo de registro com a opção `-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

```

O dig tem muitas opções que permitem refinar a saída e a consulta enviada ao servidor. Essas opções começam com `+`. Uma delas é a opção `short`, que suprime todas as saídas, exceto o resultado:

```

$ 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

```

Eis um exemplo de desativação da extensão EDNS cookie:

```

$ dig +nocookie -t MX lpi.org

; <>> DiG 9.11.5-P4-5.1+deb10u1-Debian <>> +nocookie -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
```

Exercícios Guiados

1. O que faz o comando abaixo?

```
getent group openldap
```

2. Qual a maior diferença entre o `getent` e as outras ferramentas apresentadas, `host` e `dig`?

3. Qual opção de `dig` e `host` é usada para especificar o tipo de registro que se deseja recuperar?

4. Qual das opções a seguir é uma entrada correta de `/etc/hosts`?

<code>::1 localhost</code>	
<code>localhost 127.0.0.1</code>	

5. Qual opção de `getent` é usada para especificar a fonte de dados a ser usada para realizar uma pesquisa?

Exercícios Exploratórios

1. Se o arquivo `/etc/resolv.conf` abaixo fosse alterado com um editor de texto, o que provavelmente aconteceria?

```
# Generated by NetworkManager
nameserver 192.168.1.20
```

As alterações serão sobreescritas pelo NetworkManager.

O NetworkManager atualizará sua configuração com as alterações realizadas.

As alterações não afetarão o sistema.

O NetworkManager será desabilitado.

2. O que significa a seguinte linha em `/etc/nsswitch.conf`?

```
hosts: files [SUCCESS=continue] dns
```

3. Considerando o `/etc/resolv.conf` a seguir, por que o sistema não está resolvendo nomes através do DNS?

```
search lpi.org
#nameserver fd00:ffff::1:53
#nameserver 10.0.1.53
```

4. O que faz o comando `dig +noall +answer +question lpi.org`?

5. Como passar por cima dos padrões de `dig` sem especificá-los na linha de comando?

Resumo

O comando `getent` é uma ótima ferramenta para ver os resultados das chamadas do resolvedor. Para consultas DNS simples, `host` é fácil de usar e produz uma saída direta. Se você precisa de informações detalhadas ou necessita ajustar uma consulta DNS, `dig` é provavelmente a melhor opção.

Devido à capacidade de adicionar plug-ins de bibliotecas compartilhadas e configurar o comportamento do resolvedor, o Linux tem excelente suporte para a resolução de nomes e números de vários tipos. O programa `getent` pode ser usado para resolver nomes graças às bibliotecas do resolvedor. `host` e `dig` permitem consultar os servidores DNS.

O arquivo `/etc/nsswitch.conf` é empregado para configurar o comportamento do resolvedor. Você pode alterar as fontes de dados e adicionar lógicas condicionais simples para tipos de nome com múltiplas fontes.

O DNS é configurado editando-se `/etc/resolv.conf`. Muitas distribuições têm ferramentas que gerenciam esse arquivo para você; portanto, consulte a documentação do seu sistema se as alterações manuais não persistirem.

O arquivo `/etc/hosts` é usado para resolver nomes de host para IPs e vice-versa. Ele tipicamente é empregado para definir nomes, como `localhost`, que não estão disponíveis através do DNS.

É possível deixar comentários nos arquivos de configuração abordados nesta lição. Qualquer texto à direita de `#` é ignorado pelo sistema.

Respostas aos Exercícios Guiados

1. O que faz o comando abaixo?

```
getent group openldap
```

Ele lê `/etc/nsswitch.conf`, busca pelo grupo `openldap` nas fontes listadas e exibe informações a respeito dele caso o encontre.

2. Qual a maior diferença entre o `getent` e as outras ferramentas apresentadas, `host` e `dig`?

`getent` procura por nomes usando as bibliotecas do resolvedor, os outros simplesmente consultam o DNS. O `getent` pode ser usado para a resolução de problemas em `/etc/nsswitch.conf` e para configurar as bibliotecas de resolução de nomes que seu sistema está configurado para usar. `host` e `dig` são usados para pesquisar registros de DNS.

3. Qual opção de `dig` e `host` é usada para especificar o tipo de registro que se deseja recuperar?

Ambos os programas usam `-t` para especificar o tipo de registro que se deseja consultar.

4. Qual das opções a seguir é uma entrada correta de `/etc/hosts` entry?

<code>::1 localhost</code>	X
localhost 127.0.0.1	

`::1 localhost` é a linha correta. A coluna da esquerda é sempre um endereço IPv4 ou IPv6.

5. Qual opção de `getent` é usada para especificar a fonte de dados a ser usada para realizar uma pesquisa?

A opção `-s` é usada para especificar a fonte de dados. Por exemplo:

```
$ 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
```

Respostas aos Exercícios Exploratórios

1. Se o arquivo `/etc/resolv.conf` abaixo fosse alterado com um editor de texto, o que provavelmente aconteceria?

```
# Generated by NetworkManager
nameserver 192.168.1.20
```

As alterações serão sobreescritas pelo NetworkManager.	X
O NetworkManager atualizará sua configuração com as alterações realizadas.	
As alterações não afetarão o sistema.	
O NetworkManager será desabilitado.	

2. O que significa a seguinte linha em `/etc/nsswitch.conf`?

```
hosts: files [SUCCESS=continue] dns
```

As pesquisas por nomes de host verificarão primeiro os arquivos de `/etc/hosts` e em seguida o DNS. Se uma entrada for encontrada nos arquivos e no DNS, a entrada no DNS será preferida.

3. Considerando o `/etc/resolv.conf` a seguir, por que o sistema não está resolvendo nomes através do DNS?

```
search lpi.org
#nameserver fd00:ffff::1:53
#nameserver 10.0.1.53
```

Ambos os servidores DNS estão marcados como comentários e não há nenhum servidor DNS em execução no host local.

4. O que faz o comando `dig +noall +answer +question lpi.org`?

Ele pesquisa o registro A de `lpi.org` e exibe apenas a consulta e a resposta.

5. Como passar por cima dos padrões de `dig` sem especificá-los na linha de comando?

Criando um arquivo `.digrc` em seu diretório inicial.



Tópico 110: Segurança



110.1 Tarefas administrativas de segurança

Referência ao LPI objectivo

LPIC-1 version 5.0, Exam 102, Objective 110.1

Peso

3

Áreas chave de conhecimento

- Auditar um sistema para encontrar arquivos com os bits suid/sgid ligados.
- Definir ou modificar as senhas dos usuários e as informações de expiração das senhas.
- Ser capaz de usar o nmap e o netstat para descobrir portas abertas em um sistema.
- Definir limites sobre os logins do usuário, processos e uso de memória.
- Determinar quais usuários se conectaram ao sistema ou estão conectados no momento.
- Uso e configuração básica do sudo.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- `find`
- `passwd`
- `fuser`
- `lsof`
- `nmap`
- `chage`
- `netstat`
- `sudo`

- /etc/sudoers
- su
- usermod
- ulimit
- who, w, last



**Linux
Professional
Institute**

110.1 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	110 Segurança
Objetivo:	110.1 Executar tarefas administrativas de segurança
Lição:	1 de 1

Introdução

A segurança é fundamental na administração de qualquer sistema. Como um bom administrador de sistemas Linux, você deve ficar atento a uma série de coisas, como permissões especiais em arquivos, validade das senhas de usuário, portas e sockets abertos, limitações no uso de recursos do sistema, gestão de usuários conectados e escalonamento de privilégios por meio `su` e `sudo`. Nesta lição, trataremos de cada um desses tópicos.

Verificando Arquivos com SUID e SGID

Além do conjunto de permissões tradicional de *leitura*, *escrita* e *execução*, os arquivos de um sistema Linux também podem ter algumas permissões especiais definidas, como os bits *SUID* ou *SGID*.

O bit SUID permite que o arquivo seja executado com privilégios de proprietário do arquivo. Numericamente, ele é representado por `4000` e, simbolicamente, por `s` ou `S` no bit de permissão de *execução* do proprietário. Um exemplo clássico de arquivo executável com a permissão SUID definida é `passwd`:

```
carol@debian:~$ ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 63736 jul 27 2018 /usr/bin/passwd
```

O s minúsculo em rws indica a presença do SUID no arquivo—junto com a permissão de execução. Se, no lugar dele, houvesse um S maiúsculo (rwS), isso significaria que a permissão de execução presumida não foi definida.

NOTE

Trataremos de passwd na próximo seção. O utilitário é empregado principalmente pelo root para definir/alterar as senhas de usuário (por exemplo: passwd carol). No entanto, os usuários regulares também podem usá-lo para alterar suas próprias senhas. Portanto, ele vem com o SUID definido.

Por outro lado, o bit SGID pode ser definido tanto em arquivos quanto em diretórios. Nos arquivos, seu comportamento é equivalente ao de SUID, mas os privilégios serão os mesmos do proprietário do grupo. Quando definido em um diretório, no entanto, ele permite que todos os arquivos criados ali herdem a propriedade do grupo do diretório. Como o SUID, o SGID é simbolicamente representado por s ou S no bit de permissão de execução do grupo. Numericamente, ele é representado por 2000. Para definir o SGID em um diretório, usamos chmod. É necessário adicionar 2 (SGID) às permissões tradicionais (755, em nosso caso):

```
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
```

Para encontrar arquivos com SUID e/ou SGID definidos, use o comando find com a opção -perm. Podemos incluir valores numéricos e simbólicos. Os valores—por sua vez—podem ser passados sozinhos ou precedidos por travessão (-) ou barra (/). O significado é o seguinte:

-perm numeric-value ou -perm symbolic-value

encontrar arquivos com a permissão especial *exclusivamente*

-perm -numeric-value ou -perm -symbolic-value

encontrar arquivos com a permissão especial e outras permissões

-perm /numeric-value ou -perm /symbolic-value

encontrar arquivos com qualquer uma das permissões especiais (e outras permissões)

Por exemplo, para localizar arquivos com *apenas* o SUID definido no diretório de trabalho atual,

usaríamos o seguinte comando:

```
carol@debian:~$ find . -perm 4000
carol@debian:~$ touch file
carol@debian:~$ chmod 4000 file
carol@debian:~$ find . -perm 4000
./file
```

Observe que — uma vez que não existiam arquivos contendo exclusivamente o SUID — criamos um para mostrar algumas saídas. Podemos executar o mesmo comando em notação simbólica:

```
carol@debian:~$ find . -perm u+s
./file
```

Para encontrar arquivos que correspondam ao SUID (independentemente de quaisquer outras permissões) no diretório /usr/bin/, use um dos seguintes comandos:

```
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
```

Se estiver procurando por arquivos no mesmo diretório com o bit SGID definido, execute `find /usr/bin/ -perm -2000` ou `find /usr/bin/ -perm -g+s`.

Finalmente, para encontrar arquivos com qualquer uma das duas permissões especiais definidas, adicione 4 e 2 e use /:

```
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
```

Gerenciamento e validade das senhas

Como dito acima, o utilitário `passwd` permite alterar nossa própria senha como um usuário regular. Além disso, com a opção `-S` ou `--status` podemos obter informações de status sobre nossa conta:

```
carol@debian:~$ passwd -S
carol P 12/07/2019 0 99999 7 -1
```

Eis uma análise dos sete campos obtidos na saída:

carol

Nome de login do usuário.

P

Indica que o usuário possui uma senha válida (P); outros valores possíveis são L para uma senha bloqueada e NP para nenhuma senha.

12/07/2019

Data da última alteração da senha.

0

Idade mínima em dias (o número mínimo de dias entre mudanças de senha). Um valor de 0 significa que a senha pode ser alterada a qualquer momento.

99999

Idade máxima em dias (o número máximo de dias em que a senha é válida). Um valor de 99999 desabilita a expiração da senha.

7

Período de aviso em dias (o número de dias antes da expiração da senha em que um usuário será avisado).

-1

Período de inatividade da senha em dias (o número de dias inativos após a expiração da senha até que a conta seja bloqueada). Um valor de -1 remove a inatividade de uma conta.

Além de relatar o status da conta, o comando `passwd` empregado com permissão de root serve para realizar algumas manutenções básicas de conta. Ele permite bloquear e desbloquear contas, forçar um usuário a alterar sua senha no próximo login e excluir a senha de um usuário com as opções `-l`, `-u`, `-e` e `-d`, respectivamente.

Para testar essas opções, é conveniente apresentarmos o comando `su` neste momento. Através do `su`, você pode trocar de usuário durante uma sessão de login. Assim, por exemplo, usariamos `passwd` como root para bloquear a senha de `carol`. Em seguida, mudaríamos para a conta de `carol` para verificar no status da conta se a senha foi — de fato — bloqueada (`L`) e não pode ser alterada. Finalmente, voltando ao usuário root, desbloqueamos a senha de `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.
```

Outra alternativa para bloquear e desbloquear a senha de um usuário é o comando `usermod`:

Bloquear a senha da usuária carol

```
usermod -L carol or usermod --lock carol.
```

Desbloquear a senha da usuária carol

```
usermod -U carol or usermod --unlock carol.
```

NOTE Com as opções `-f` ou `--inactive`, `usermod` também serve para definir o número de dias até que uma conta com a senha expirada seja desabilitada (p. ex. `usermod -f 3 carol`).

Além de `passwd` e `usermod`, o comando mais direto para lidar com senhas e validade de contas é `chage` (“change age”, ou alterar idade). Como root, você pode passar para `chage` a opção `-l` (ou `--list`) seguida por um nome de usuário para imprimir na tela a senha atual desse usuário e as informações de expiração da conta; como usuário comum, você pode ver suas próprias informações:

```
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
```

Quando executado sem opções e seguido apenas por um nome de usuário, `chage` se comporta de maneira interativa:

```
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]:
```

As opções para modificar as diferentes configurações de chage são as seguintes:

-m days username ou --mindays days username

Especifica o número mínimo de dias entre as alterações de senha (por exemplo: chage -m 5 carol). Um valor de 0 permite que o usuário altere sua senha a qualquer momento.

-M days username ou --maxdays days username

Especifica o número máximo de dias em que a senha será válida (por exemplo: chage -M 30 carol). Para desabilitar a expiração da senha, é comum atribuir a esta opção o valor 99999.

-d days username ou --lastday days username

Especifica o número de dias desde que a senha foi alterada pela última vez (por exemplo: chage -d 10 carol). Um valor de 0 força o usuário a alterar sua senha no próximo login.

-W days username ou --warndays days username

Especifica o número de dias em que o usuário será lembrado de que sua senha expirou.

-I days username ou --inactive days username

Especifica o número de dias de inatividade após a expiração da senha (por exemplo: chage -I 10 carol)—o mesmo que usermod -f ou usermod --inactive. Transcorridos esses dias, a conta será bloqueada. Porém, se o valor for 0, a conta não será bloqueada.

-E date username ou --expiredate date username

Especifica a data (ou número de dias desde a época—1º de janeiro de 1970) em que a conta será bloqueada. Normalmente é expresso no formato AAAA-MM-DD (por exemplo: chage -E 2050-12-13 carol).

NOTE

Para aprender mais sobre passwd, usermod e chage — e suas opções — consulte as páginas de manual respectivas.

Descobrindo portas abertas

Os administradores de sistema também precisam ficar de olho nas portas abertas e, para isso, existem quatro utilitários muito poderosos na maioria dos sistemas Linux: lsof, fuser, netstat e nmap. Falaremos deles nesta seção.

lsof significa “list open files”, o que não é pouca coisa, já que — para o Linux — tudo é arquivo.

Na verdade, se você digitar `lsof` no terminal, aparecerá uma enorme lista de arquivos comuns, arquivos de dispositivos, sockets etc. Porém, nesta lição, vamos nos concentrar principalmente nas portas. Para imprimir a lista de todos os arquivos de rede da “Internet”, execute `lsof` com a opção `-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)
```

Além do serviço `bootpc`—que é usado pelo DHCP—a saída mostra dois serviços ouvindo conexões—`ssh` e o servidor web Apache (`http`)—bem como duas conexões SSH estabelecidas. Podemos especificar um determinado host com a notação `@ip-address` para verificar suas conexões:

```
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

Para imprimir apenas os arquivos de rede IPv4 e IPv6, use as opções `-i4` e `-i6`, respectivamente.

Da mesma forma, é possível filtrar por porta, passando para a opção `-i` (ou `-i@ip-address`) o argumento `: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)
```

Ao incluir mais de uma porta, nós as separamos com vírgulas (os intervalos são especificados com um traço):

```
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

O `lsof` tem uma quantidade impressionante de opções disponíveis. Para saber mais, consulte a página de manual.

O próximo na lista de comandos de rede é `fuser`. Seu propósito principal é encontrar um “usuário do arquivo” — o que envolve saber quais processos estão acessando quais arquivos; ele também fornece algumas outras informações, como o tipo de acesso. Por exemplo, para verificar o diretório de trabalho atual, basta executar `fuser ..`. Mas quando precisamos de mais informações, é aconselhável usar a opção verbose (`-v` ou `--verbose`):

```
root@debian:~# fuser .
/root:                580c
root@debian:~# fuser -v .
                      USER      PID ACCESS COMMAND
/root:                root      580  ...c... bash
```

Vamos detalhar a saída:

Arquivo

O arquivo sobre o qual estamos obtendo informações (`/root`).

Coluna USER

O proprietário do arquivo (`root`).

Coluna PID

O identificador do processo (580).

Coluna ACCESS

Tipo de acesso (.. c ..). Ele pode ser:

c

Diretório atual.

e

Executável em execução.

f

Arquivo aberto (omitido no modo de exibição padrão).

F

Arquivo aberto para escrita (omitido no modo de exibição padrão).

r

Diretório raiz.

m

Arquivo mmap ou biblioteca compartilhada.

.

Espaço reservado (omitido no modo de exibição padrão).

Coluna COMMAND

O comando afiliado ao arquivo (bash).

Com a opção -n (ou --namespace), podemos encontrar informações sobre portas/sockets de rede. Também é necessário fornecer o protocolo de rede e o número da porta. Assim, para obter informações sobre o servidor web Apache, o comando seria o seguinte:

```
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 também pode ser usado para eliminar os processos que estão acessando o arquivo, com as opções -k ou --kill (p. ex.: fuser -k 80/tcp). Consulte a página de manual para saber mais detalhes.

Vamos olhar o `netstat` agora. O `netstat` é uma ferramenta de rede muito versátil usada principalmente para imprimir “estatísticas de rede”.

Executado sem opções, `netstat` exibe as conexões ativas da Internet e os sockets Unix. Devido ao tamanho da lista, vale a pena canalizar a saída para `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
(...)
```

Para listar apenas as portas e sockets “ouvintes”, usamos as opções `-l` ou `--listening`. As opções `-t`/`--tcp` e `-u`/`--udp` podem ser adicionadas para filtrar pelos protocolos TCP e UDP, respectivamente (elas também podem ser combinadas no mesmo comando). Da mesma forma, ``-e``/`--extend` exibe informações adicionais:

```
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:*
tcp      0      0 localhost:smtp          0.0.0.0:*
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:*
13729
tcp      0      0 localhost:smtp          0.0.0.0:*
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:*        root
13604

```

Se você omitir a opção `-l`, *apenas* as conexões estabelecidas serão mostradas:

```

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

```

Se você está interessado apenas em informações numéricas sobre portas e hosts, pode usar a opção `-n` or `--numeric` para imprimir apenas os números de portas e endereços IP. Note como `ssh` se transforma em `22` ao adicionarmos `-n` ao comando acima:

```

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

```

Como vemos, é possível tornar os comandos `netstat` muito úteis e produtivos combinando algumas de suas opções. Navegue pelas páginas de manual para aprender mais e encontrar as combinações que melhor atendem às suas necessidades.

Finalmente, vamos falar do `nmap` — ou “mapeador de rede”. Poderosíssimo, este scanner de porta é executado especificando-se um endereço IP ou nome de host:

```

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

```

Além de um host único, o `nmap` permite escanear:

Múltiplos hosts

separando-os com espaços (p. ex.: `nmap localhost 192.168.1.7`).

Intervalos de hosts

usando um traço (p. ex.: `nmap 192.168.1.3-20`).

subredes

usando um curinga ou notação CIDR (p. ex.: `nmap 192.168.1.*` ou `nmap 192.168.1.0/24`). Podemos excluir determinados hosts (p. ex.: `nmap 192.168.1.0/24 --exclude 192.168.1.7`).

Para escanear uma porta específica, use a opção `-p` seguida pelo número da porta ou nome do serviço (`nmap -p 22` e `nmap -p ssh` terão a mesma saída):

```

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

```

Também podemos escanear várias portas ou intervalos de portas usando vírgulas e travessões, respectivamente:

```

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

```

Duas outras opções importantes e úteis do `nmap` são:

-F

Executar um escaneamento rápido das 100 portas mais comuns.

-v

Obter uma saída detalhada (-vv imprime uma saída mais detalhada ainda).

NOTE O `nmap` é capaz de executar comandos bastante complexos usando vários tipos de escaneamento. No entanto, esse tópico está fora do escopo desta lição.

Limites em logins de usuário, processos e uso de memória

Os recursos de um sistema Linux não são ilimitados, e portanto—como administrador do sistema—você deve garantir um bom equilíbrio entre os *limites do usuário* no uso dos recursos e o funcionamento adequado do sistema operacional. O `ulimit` pode ajudá-lo nesse aspecto.

O `ulimit` lida com os limites *soft* (flexíveis) e *hard* (rígidos)—especificados pelas opções `-S` e `-H`, respectivamente. Quando executado sem opções ou argumentos, o `ulimit` exibe os blocos de arquivos flexíveis do usuário atual:

```
carol@debian:~$ ulimit
```

unlimited

Com a opção `-a`, o `ulimit` mostra todos os limites flexíveis atuais (o mesmo que `-Sa`); para exibir todos os limites rígidos atuais, use `-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
(...)
```

Os recursos de shell disponíveis são especificados por opções como:

-b

Tamanho máximo do buffer do socket

-f

tamanho máximo dos arquivos escritos pelo shell e seus processos-filhos

-l

tamanho máximo da memória que pode ser bloqueada

-m

tamanho máximo do conjunto residente (RSS, resident set size) — a porção atual da memória mantida por um processo na memória principal (RAM)

-v

quantidade máxima de memória virtual

-u

número máximo de processos disponíveis para um único usuário

Assim, para exibir os limites, usariamos `ulimit` seguido por `-S` (soft) ou `-H` (hard) e a opção referente ao recurso; se nenhum `-S` ou `-H` for fornecido, os limites flexíveis (soft) serão mostrados:

```
carol@debian:~$ ulimit -u
10000
carol@debian:~$ ulimit -Su
10000
carol@debian:~$ ulimit -Hu
15672
```

Da mesma maneira, para definir novos limites para um recurso determinado, especificamos **-S** ou **-H** seguidos pela opção de recurso desejada e o novo valor. Esse valor pode ser um número ou as palavras especiais **soft** (limite flexível atual), **hard** (limite rígido atual) ou **unlimited** (sem limite). Se nem **-S** nem **-H** forem especificados, ambos os limites serão definidos. Por exemplo, vamos primeiro ler o valor do tamanho máximo atual para os arquivos gravados pelo shell e seus filhos:

```
root@debian:~# ulimit -Sf
unlimited
root@debian:~# ulimit -Hf
unlimited
```

Agora, vamos alterar o valor de **unlimited** para **500** sem especificar **-S** ou **-H**. Observe como os limites flexíveis e rígidos são alterados:

```
root@debian:~# ulimit -f 500
root@debian:~# ulimit -Sf
500
root@debian:~# ulimit -Hf
500
```

Por fim, diminuímos apenas o limite flexível para blocos de **200**:

```
root@debian:~# ulimit -Sf 200
root@debian:~# ulimit -Sf
200
root@debian:~# ulimit -Hf
500
```

Os limites rígidos só podem ser aumentados pelo usuário root. Por outro lado, os usuários comuns podem diminuir os limites rígidos e aumentar os limites flexíveis até o valor dos limites rígidos. Para tornar persistentes os novos valores de limites após a reinicialização, eles devem ser

gravados no arquivo `/etc/security/limits.conf`. Este também é o arquivo usado pelo administrador para aplicar restrições a usuários específicos.

NOTE Saiba que não existe uma página de manual para `ulimit`. Trata-se de uma ferramenta nativa do bash e, por isso, é preciso consultar a man page do bash para saber mais sobre ela.

Lidando com usuários conectados

Outra de suas funções como administrador do sistema envolve manter o controle dos usuários conectados. Há três utilitários que podem ajudá-lo com essas tarefas: `last`, `who` e `w`.

O `last` imprime uma lista dos últimos usuários logados, com as informações mais recentes no topo:

```
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
```

Considerando a listagem truncada, obtemos informações sobre os dois últimos usuários do sistema. As duas primeiras linhas trazem informações sobre a usuária `carol`; as duas linhas seguintes, sobre a usuária `mimi`. As informações são as seguintes:

1. A usuária `carol`, no terminal `pts/0` do host `192.168.1.4`, iniciou sua sessão em `Sat Jun 6` (sábado, 6 de junho) às `14:25` e está `still logged in` (ainda logada). O sistema — usando o kernel `4.19.0-9-amd64` — foi iniciado (`reboot system boot`) em `Sat Jun 6` (sábado, 6 de junho) às `14:24` e está `still running` (ainda em execução).
2. A usuária `mimi`, no terminal `pts/0` do host `192.168.1.4`, iniciou sua sessão em `Sat Jun 6` (sábado, 6 de junho) às `12:07` e se desconectou às `14:24` (a sessão durou um total de `(02:16)` horas). O sistema — usando o kernel `4.19.0-9-amd64` — foi iniciado (`reboot system boot`) em `Sat Jun 6` (sábado, 6 de junho) às `12:07` e desligado às `14:24` (tendo estado ativo por `(02:17)` horas).

NOTE A linha `wtmp begins Sun May 31 14:14:58 2020` se refere a `/var/log/wtmp`, o arquivo especial de log no qual `last` obtém as informações.

Podemos passar um nome de usuário a `last` para ver apenas as entradas referentes a ele:

```
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)
(...)
```

Em relação à segunda coluna (terminal), pts significa *Pseudo Terminal Slave* — em oposição a um terminal *TeleTYewriter* ou tty; 0 refere-se ao primeiro (a contagem começa em zero).

NOTE Para verificar se há tentativas de login incorretas, execute `lastb` em vez de `last`.

Os utilitários `who` e `w` concentram-se nos usuários atualmente logados e são bastante semelhantes. O primeiro exibe quem está conectado, ao passo que o último também exibe informações sobre o que eles estão fazendo.

Quando executado sem opções, o `who` exibe quatro colunas correspondentes ao usuário conectado, terminal, data e hora e nome do host:

```
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)
```

O `who` aceita uma série de opções, dentre as quais podemos destacar as seguintes:

-b,--boot

Exibe a hora da última inicialização do sistema.

-r,--runlevel

Mostra o nível de execução atual.

-H,--heading

Imprime o cabeçalho das colunas.

Comparado ao `who`, o `w` fornece uma saída um pouco mais detalhada:

```
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
```

A linha superior fornece informações sobre a hora atual (17:56:12), há quanto tempo o sistema está ativo (up 40 min), o número de usuários atualmente logados (2 users) e os números médios de carga (load average: 0.04, 0.12, 0.09). Esses valores referem-se ao número médio de trabalhos na fila de execução nos últimos 1, 5 e 15 minutos, respectivamente.

Vêm em seguida oito colunas; elas se dividem desta forma:

USER

Nome de login do usuário.

TTY

Nome do terminal em que o usuário está.

FROM

Host remoto a partir do qual o usuário se logou.

LOGIN@

Tempo de login.

IDLE

Tempo ocioso.

JCPU

Tempo usado por todos os processos ligados ao tty (incluindo os trabalhos atualmente em execução em segundo plano).

PCPU

Tempo usado pelo processo atual (que aparece sob **WHAT**).

WHAT

Linha de comando do processo atual.

Como no caso do who, podemos passar nomes de usuários ao 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
```

Configuração e uso básicos do sudo

Como já observado nesta lição, `su` permite alternar para qualquer outro usuário no sistema, desde que você forneça a senha do usuário de destino. No caso do usuário root, ter sua senha distribuída ou conhecida por (muitos) usuários põe o sistema em risco e é uma péssima prática de segurança. O uso básico de `su` é `su - nome-de-usuário`. Ao alternar para root, porém, o nome de usuário de destino é opcional:

```
carol@debian:~$ su - root
Password:
root@debian:~# exit
logout
carol@debian:~$ su -
Password:
root@debian:~#
```

O uso do traço (-) garante que o ambiente do usuário de destino seja carregado. Sem ele, será mantido o ambiente do antigo usuário:

```
carol@debian:~$ su
Password:
root@debian:/home/carol#
```

Por outro lado, existe o comando `sudo`. Com ele, é possível executar um comando como usuário root—ou qualquer outro usuário. Do ponto de vista da segurança, `sudo` é uma opção muito melhor do que `su`, pois apresenta duas vantagens principais:

1. para executar um comando como root, você não precisa da senha do usuário root, mas apenas a do usuário atual, em conformidade com uma política de segurança. A política de segurança padrão é `sudoers`, especificada em `/etc/sudoers` e `/etc/sudoers.d/*`.
2. O `sudo` permite executar comandos simples com privilégios elevados em vez de lançar um novo subshell para root, como faria `su`.

O uso básico de `sudo` é `sudo -u target-username command`. Porém, para executar um comando como usuário root, a opção `-u target-username` não é necessária:

```
carol@debian:~$ sudo -u mimi whoami
mimi
carol@debian:~$ sudo whoami
```

root

NOTE

O sudoers usa uma marca temporal por usuário (e por terminal) para ocultar as credenciais, sendo assim possível usar o sudo sem uma senha por um período padrão de quinze minutos. Esse valor padrão pode ser modificado adicionando-se a opção timestamp_timeout como configuração de Defaults em /etc/sudoers (p. ex.: Defaults timestamp_timeout=1 define o tempo limite do cache de credenciais como um minuto).

O arquivo /etc/sudoers

O arquivo de configuração principal do sudo é /etc/sudoers (também existe o diretório /etc/sudoers.d). É ali que os privilégios de sudo dos usuários são determinados. Em outras palavras, aqui você especifica quem pode executar quais comandos, sob quais nomes de usuário e em quais máquinas — bem como outras configurações. A sintaxe usada é a seguinte:

```
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
(...)
```

A especificação de privilégio para o usuário root é ALL=(ALL:ALL) ALL. Isso se traduz assim: o usuário root (root) pode se logar com todos os hosts (ALL), em nome de todos os usuários e todos os grupos ((ALL:ALL)), além de executar todos os comandos (ALL). O mesmo vale para os membros do grupo sudo — note como os nomes de grupos são identificados com um símbolo de porcentagem (%).

Assim, para que a usuária carol seja capaz de verificar o status de apache2 de qualquer host em nome de qualquer usuário ou grupo, adicionariámos a seguinte linha ao arquivo sudoers:

```
carol    ALL=(ALL:ALL) /usr/bin/systemctl status apache2
```

Para evitar a carol o incômodo de precisar fornecer sua senha para executar o comando systemctl status apache2, modificamos a linha desta maneira:

```
carol    ALL=(ALL:ALL) NOPASSWD: /usr/bin/systemctl status apache2
```

Digamos que agora você queira restringir seus hosts a 192.168.1.7 e permitir que carol execute systemctl status apache2 no nome da usuária mimi. A linha teria de ser modificada desta forma:

```
carol    192.168.1.7=(mimi) /usr/bin/systemctl status apache2
```

Vamos então verificar o status do servidor web Apache como a usuária 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
(...)
```

Se carol fosse promovida a sysadmin e você quisesse conceder a ela todos os privilégios, a maneira mais fácil seria incluí-la no grupo especial sudo com usermod e a opção -G (e talvez também a opção -a, que garante que o usuário não seja removido de nenhum outro grupo ao qual possa pertencer):

```
root@debian:~# sudo useradd -aG sudo carol
```

NOTE

Na família de distribuições Red Hat, o grupo wheel é equivalente ao grupo especial de administradores sudo dos sistemas Debian.

Em vez de editar /etc/sudoers diretamente, use simplesmente o comando visudo como root (p. ex. visudo), que abre /etc/sudoers usando o editor de texto predefinido. Para alterar o editor de texto padrão, adicione a opção editor como configuração de Defaults em /etc/sudoers. Assim, por exemplo, para mudar o editor para nano, a linha seria a seguinte:

```
Defaults    editor=/usr/bin/nano
```

NOTE

Outra alternativa é especificar um editor de texto por meio da variável de ambiente EDITOR usando visudo (e.g.: EDITOR=/usr/bin/nano visudo)

Além dos usuários e grupos, também podemos utilizar aliases em /etc/sudoers. É possível definir três categorias principais de aliases: *aliases de host* (Host_Alias), *aliases de usuário*

(User_Alias) e *aliases de comando* (Cmnd_Alias). Eis um exemplo:

```
# 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
```

A partir deste arquivo `sudoers` de amostra, vamos explicar os três tipos de aliases com um pouco mais de detalhes:

Aliases de host

Incluem uma lista separada por vírgulas de nomes de host, endereços IP, redes e grupos de rede (precedidos por `+`). Máscaras de rede também podem estar especificadas. O alias de host `SERVERS` inclui um endereço IP e dois nomes de host:

```
Host_Alias SERVERS = 192.168.1.7, server1, server2
```

Aliases de usuário

Incluem uma lista separada por vírgulas de usuários especificados como nomes de usuários, grupos (precedidos por `%`) e grupos de rede (precedidos por `+`). Para excluir usuários específicos, usamos `!`. O alias de usuário `ADMINS`—por exemplo—inclui a usuária `carol`, os membros do grupo `sudo` e os membros do alias de usuário `PRIVILEGE_USERS` que não pertencem ao alias de usuário `REGULAR_USERS`:

```
User_Alias ADMIN = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS
```

Aliases de comando

Incluem uma lista separada por vírgulas de comandos e diretórios. Se um diretório for especificado, qualquer arquivo que esteja nesse diretório será incluído — mas os subdiretórios serão ignorados. O alias do comando SERVICES inclui um único comando com todos os seus subcomandos — conforme especificado pelo asterisco (*):

```
Cmnd_Alias SERVICES = /usr/bin/systemctl *
```

Como resultado das especificações de alias, a linha ADMIN SERVERS=SERVICES na seção User privilege specification é traduzida como: todos os usuários pertencentes a ADMIN podem usar sudo para executar qualquer comando em SERVICES em qualquer servidor em SERVERS.

NOTE

Existe um quarto tipo de alias que pode ser incluído em /etc/sudoers: *aliases de execução* (Runas_Alias). São muito semelhantes aos aliases de usuário, mas permitem especificar usuários por seu *ID de usuário* (UID). Esse recurso pode ser conveniente em certos casos.

Exercícios Guiados

1. Preencha a seguinte tabela com relação às permissões especiais::

Permissão especial	Representação Numérica	Representação Simbólica	Encontrar arquivos com <i>somente esse conjunto de permissões</i>
SUID			
SGID			

2. Exibir arquivos com *somente* o conjunto de bits SUID ou SGID normalmente não é muito prático. Execute as seguintes tarefas para demonstrar que suas buscas podem ser mais produtivas:

- Encontre todos os arquivos com o SUID (e outras permissões) definido em /usr/bin:

- Encontre todos os arquivos com o SGID (e outras permissões) definido em /usr/bin:

- Encontre todos os arquivos com SUID ou SGID definido em /usr/bin:

3. O chage permite alterar as informações de expiração de senha de um usuário. Como root, complete a seguinte tabela, fornecendo os comandos corretos para o usuário mary:

Significado	Comandos chage
Faça a senha ser válida por 365 dias.	
Faça o usuário alterar a senha no próximo login.	
Defina o número mínimo de dias entre as alterações de senha para 1.	
Desative a expiração da senha.	
Permita que o usuário altere sua senha a qualquer momento.	

Significado	Comandos chage
Defina o período de aviso para 7 dias e a data de expiração da conta para 20 de agosto de 2050.	
Imprima as informações de validade da senha atual do usuário.	

4. Preencha a seguinte tabela com o utilitário de rede apropriado:

Ação	Comando(s)
Mostrar os arquivos de rede do host 192.168.1.55 na porta 22 usando lsof.	
Mostrar os processos que acessam a porta padrão do servidor web Apache em sua máquina com fuser.	
Listar todos os sockets <i>udp</i> ouvintes em sua máquina usando netstat.	
Escanear as portas 80 até 443 no host 192.168.1.55 usando nmap.	

5. Execute as seguintes tarefas relativas ao *tamanho do conjunto residente (RSS)* e *ulimit* como um usuário regular:

- Exiba os limites *flexíveis* do *RSS máximo*:

- Exiba os limites *rígidos* do *RSS máximo*:

- Defina os limites *flexíveis* do *RSS máximo* para 5.000 kilobytes:

- Defina os limites *rígidos* do *RSS máximo* para 10.000 kilobytes:

- Finalmente, tente aumentar o limite *rígido* do *RSS máximo* para 15.000 kilobytes. Você conseguiu? Por quê?

6. Considere a seguinte linha de saída de comando `last` e responda às perguntas:

```
carol      pts/0        192.168.1.4        Sun May 31 14:16 - 14:22 (00:06)
```

- `carol` estava conectada a partir de um host remoto? Por quê?

- Quanto tempo durou a sessão de `carol`?

- `carol` estava conectada através de um terminal de texto clássico? Por quê?

7. Analise o seguinte trecho de `/etc/sudoers` e responda à questão abaixo.

```
# 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` pode verificar o status do servidor web Apache em qualquer host? Por quê?

Exercícios Exploratórios

1. Além de SUID e SGID, existe uma terceira permissão especial: o *sticky bit*. No momento, ele é usado principalmente em diretórios como /tmp para evitar que usuários regulares excluam ou movam arquivos que não sejam seus. Realize as seguintes tarefas:

- Defina o *sticky bit* em ~/temporal:

- Encontre diretórios com o *sticky bit* (e quaisquer outras permissões) definidas em seu diretório inicial:

- Remova o *sticky bit* de ~/temporal:

2. Quando a senha de um usuário é bloqueada via `passwd -l username` ou `usermod -L username`, como é possível saber disso olhando em /etc/shadow?

3. Qual o equivalente do comando `usermod` para `chage -E date username` ou `chage --expiredate date username`?

4. Forneça dois comandos `nmap` diferentes para escanear todas as 65535 no host local:

Resumo

Nesta lição, você aprendeu a executar diversas tarefas de administração ligadas à segurança. Os seguintes tópicos foram abordados:

- Encontrar arquivos com as permissões especiais SUID e SGID definidas.
- Definir e alterar as senhas dos usuários e lidar com as informações de validade da senha.
- Usar diversos utilitários de rede para encontrar portas abertas em hosts/redes.
- Configurar limites nos recursos do sistema.
- Verificar os usuários que se logaram ao sistema ou que estão atualmente logados.
- Uso e configuração básica de sudo (através do arquivo /etc/sudoers).

Comandos e arquivos discutidos nesta lição:

find

Buscar arquivos em uma hierarquia de diretórios.

passwd

Trocar a senha de usuário.

chmod

Alterar os bits de modo do arquivo.

chage

Alterar as informações de expiração da senha de usuário.

lsof

Listar os arquivos abertos.

fuser

Identificar os processos que estão usando arquivos ou sockets.

netstat

Exibir as conexões de rede.

nmap

Ferramenta de exploração de rede e scanner de portas.

ulimit

Obter e definir os limites do usuário.

/etc/security/limits.conf

Arquivo de configuração para aplicar restrições a usuários.

last

Imprimir uma lista dos últimos usuários logados.

lastb

Imprimir uma lista de tentativas de login incorretas.

/var/log/wtmp

Banco de dados de logins de usuários.

who

Mostrar quem está logado.

w

Mostrar quem está logado e o que estão fazendo.

su

Trocar usuário ou entrar como superusuário.

sudo

Executar um comando como outro usuário (incluindo o superusuário).

/etc/sudoers

Arquivo de configuração padrão para a política de segurança de **sudo**.

Respostas aos Exercícios Guiados

1. Preencha a seguinte tabela com relação às permissões especiais::

Permissão especial	Representação Numérica	Representação Simbólica	Encontrar arquivos com <i>somente esse conjunto de permissões</i>
SUID	4000	s,S	find -perm 4000, find -perm u+s
SGID	2000	s,S	find -perm 2000, find -perm g+s

2. Exibir arquivos com *somente* o conjunto de bits SUID ou SGID normalmente não é muito prático. Execute as seguintes tarefas para demonstrar que suas buscas podem ser mais produtivas:

- Encontre todos os arquivos com o SUID (e outras permissões) definido em /usr/bin:

```
find /usr/bin -perm -4000 or find /usr/bin -perm -u+s
```

- Encontre todos os arquivos com o SGID (e outras permissões) definido em /usr/bin:

```
find /usr/bin -perm -2000 or find /usr/bin -perm -g+s
```

- Encontre todos os arquivos com SUID ou SGID definido em /usr/bin:

```
find /usr/bin -perm /6000
```

3. O chage permite alterar as informações de expiração de senha de um usuário. Como root, complete a seguinte tabela, fornecendo os comandos corretos para o usuário mary:

Significado	Comandos chage
Faça a senha ser válida por 365 dias.	chage -M 365 mary, chage --maxdays 365 mary
Faça o usuário alterar a senha no próximo login.	chage -d 0 mary, chage --lastday 0 mary
Defina o número mínimo de dias entre as alterações de senha para 1.	chage -m 1 mary, chage --mindays 1 mary

Significado	Comandos chage
Desative a expiração da senha.	chage -M 99999 mary, chage --maxdays 99999 mary
Permita que o usuário altere sua senha a qualquer momento.	chage -m 0 mary, chage --mindays 0 mary
Defina o período de aviso para 7 dias e a data de expiração da conta para 20 de agosto de 2050.	chage -W 7 -E 2050-08-20 mary, chage --warndays 7 --expiredate 2050-08-20 mary
Imprima as informações de validade da senha atual do usuário.	chage -l mary, chage --list mary

4. Preencha a seguinte tabela com o utilitário de rede apropriado:

Ação	Comando(s)
Mostrar os arquivos de rede do host 192.168.1.55 na porta 22 usando lsof.	lsof -i@192.168.1.55:22
Mostrar os processos que acessam a porta padrão do servidor web Apache em sua máquina com fuser.	fuser -vn tcp 80, fuser --verbose --namespace tcp 80
Listar todos os sockets <i>udp</i> ouvintes em sua máquina usando netstat.	netstat -lu, netstat --listening --udp
Escanear as portas 80 até 443 no host 192.168.1.55 usando nmap.	nmap -p 80-443 192.168.1.55

5. Execute as seguintes tarefas relativas ao *tamanho do conjunto residente (RSS)* e ulimit como um usuário regular:

- Exiba os limites *flexíveis* do RSS *máximo*:

```
ulimit -m, ulimit -Sm
```

- Exiba os limites *rígidos* do RSS *máximo*:

```
ulimit -Hm
```

- Defina os limites *flexíveis* do RSS *máximo* para 5.000 kilobytes:

```
ulimit -Sm 5000
```

- Defina os limites *rígidos* do RSS *máximo* para 10.000 kilobytes:

```
ulimit -Hm 10000
```

- Finalmente, tente aumentar o limite *rígido* do RSS *máximo* para 15.000 kilobytes. Você conseguiu? Por quê?

Não. Uma vez definido, os usuários comuns não podem aumentar os limites rígidos.

6. Considere a seguinte linha de saída de comando last e responda às perguntas:

carol	pts/0	192.168.1.4	Sun May 31 14:16 - 14:22 (00:06)
-------	-------	-------------	----------------------------------

- `carol` estava conectada a partir de um host remoto? Por quê?

Sim, o endereço IP do host remoto está na terceira coluna.

- Quanto tempo durou a sessão de `carol`?

Seis minutos (como mostrado na última coluna).

- `carol` estava conectada através de um terminal de texto clássico? Por quê?

Não, `pts/0` na segunda coluna indica que a conexão foi feita por meio de um emulador de terminal gráfico (ou *Pseudo Terminal Slave*).

7. Analise o seguinte trecho de /etc/sudoers e responda à questão abaixo.

```
# 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 pode verificar o status do servidor web Apache em qualquer host? Por quê?

Não, pois ele é membro de REGULAR_USERS e esse grupo de usuários está excluído de ADMINS, que são os únicos usuários (além de carol, dos membros do grupo sudo e de root) que podem executar systemctl status apache2 nos SERVERS.

Respostas aos Exercícios Exploratórios

1. Além de SUID e SGID, existe uma terceira permissão especial: o *sticky bit*. No momento, ele é usado principalmente em diretórios como /tmp para evitar que usuários regulares excluam ou movam arquivos que não sejam seus. Realize as seguintes tarefas:

- Defina o *sticky bit* em ~/temporal:

```
chmod +t temporal, chmod 1755 temporal
```

- Encontre diretórios com o *sticky bit* (e quaisquer outras permissões) definidas em seu diretório inicial:

```
find ~ -perm -1000, find ~ -perm /1000
```

- Remova o *sticky bit* de ~/temporal:

```
chmod -t temporal, chmod 0755 temporal
```

2. Quando a senha de um usuário é bloqueada via `passwd -l username` ou `usermod -L username`, como é possível saber disso olhando em /etc/shadow?

Um ponto de exclamação aparecerá no segundo campo, logo após o nome de login do usuário afetado (p. ex.: mary: !\$6\$g0g9xJgv...).

3. Qual o equivalente do comando `usermod` para `chage -E date username` ou `chage --expiredate date username`?

```
usermod -e date username, usermod --expiredate date username
```

4. Forneça dois comandos `nmap` diferentes para escanear todas as 65535 no host local:

```
nmap -p 1-65535 localhost e nmap -p- localhost
```



110.2 Configurar a segurança do host

Referência ao LPI objectivo

[LPIC-1 version 5.0, Exam 102, Objective 110.2](#)

Peso

3

Áreas chave de conhecimento

- Saber que existem senhas sombreadas (shadow) e como elas funcionam.
- Desligar os serviços de rede que não estão em uso.
- Entender a função do TCP wrappers.

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- /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



**Linux
Professional
Institute**

110.2 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	110 Segurança
Objetivo:	110.2 Configurar a segurança do host
Lição:	1 de 1

Introdução

Este capítulo explica quatro maneiras básicas de melhorar a segurança do host:

1. Alguns comandos e definições de configuração básicas para aperfeiçoar a segurança da autenticação com senhas ocultas.
2. Como usar superdaemons para ouvir conexões de rede de entrada.
3. Verificação dos serviços de rede em busca de daemons desnecessários.
4. Uso de TCP wrappers como uma espécie de firewall simples.

Melhorar a segurança da autenticação com senhas ocultas

Os componentes básicos dos dados da conta de um usuário são armazenados no arquivo `/etc/passwd`. Este arquivo contém sete campos: nome de login, identidade do usuário, identidade do grupo, senha, comentário (também conhecido como GECOS), localização do diretório inicial e, finalmente, o shell padrão. Uma maneira simples de lembrar a ordem desses campos é pensar sobre o processo de login de um usuário: primeiro você insere um nome de login, em segundo lugar o sistema o mapeia em um ID de usuário (uid) e depois em um ID de grupo (gid). O quarto

passo pede uma senha, o quinto consulta o comentário, o sexto entra no diretório inicial do usuário e o sétimo configura o shell padrão.

Porém, nos sistemas modernos, a senha não é mais armazenada no arquivo `/etc/passwd`. Em vez disso, o campo de senha contém apenas um `x` minúsculo. O arquivo `/etc/passwd` precisa ser legível por todos os usuários; portanto, não é uma boa ideia armazenar senhas nele. O `x` indica que a senha criptografada (com hash) está na verdade armazenada no arquivo `/etc/shadow`. Este arquivo não deve ser legível para todos os usuários.

As configurações de senha são definidas com os comandos `passwd` e `chage`. Ambos os comandos alteram a entrada referente à usuária `emma` no arquivo `/etc/shadow`. Como superusuário, você pode definir a senha da usuária `emma` com o seguinte comando:

```
$ sudo passwd emma
New password:
Retype new password:
passwd: password updated successfully
```

Em seguida, o sistema pede que a nova senha seja confirmada duas vezes.

Para listar o tempo de expiração da senha e outras configurações de expiração de senha para a usuária `emma`, use:

```
$ 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
```

Para evitar que a usuária `emma` faça login no sistema, o superusuário pode definir uma data de expiração da senha anterior à data atual. Por exemplo, se a data de hoje fosse 27/03/2020, poderíamos definir uma data mais antiga para a expiração da senha da usuária:

```
$ sudo chage -E 2020-03-26 emma
```

Como alternativa, o superusuário poderia usar:

```
$ sudo passwd -l emma
```

para bloquear a conta temporariamente através da opção `-l` em `passwd`. Para testar os efeitos dessas alterações, tente fazer o login como a conta `emma`:

```
$ sudo login emma
Password:
Your account has expired; please contact your system administrator

Authentication failure
```

Para evitar que todos os usuários, exceto o usuário `root`, se loguem no sistema temporariamente, o superusuário pode criar um arquivo chamado `/etc/nologin`. Esse arquivo pode conter uma mensagem para os usuários notificando-os sobre o motivo pelo qual não podem fazer login no momento (por exemplo, notificações de manutenção do sistema). Para saber mais, consulte `man 5 nologin`. Note que também existe um comando `nologin` que pode ser usado para impedir um login quando definido como o shell padrão de um usuário. Por exemplo:

```
$ sudo usermod -s /sbin/nologin emma
```

Consulte `man 8 nologin` para saber mais.

Como usar um superdaemon para ouvir conexões de rede de entrada

Os serviços de rede, como os servidores web, servidores de email e servidores de impressão, geralmente funcionam como um serviço autônomo que escuta em uma porta de rede dedicada. Todos esses serviços autônomos são executados lado a lado. No sistema clássico baseado em `Sys-V-init`, cada um desses serviços pode ser controlado pelo comando `service`. Nos sistemas atuais baseados em `systemd`, usamos `systemctl` para gerenciar o serviço.

Antigamente, a disponibilidade de recursos do computador era muito menor. Não era aconselhável executar muitos serviços em modo autônomo ao mesmo tempo. Em vez disso, um superdaemon era usado para escutar as conexões de rede de entrada e iniciar o serviço apropriado sob demanda. Este método de construção de uma conexão de rede era um pouco mais demorado. `inetd` e `xinetd` são superdaemons bastante conhecidos. Nos sistemas atuais baseados em `systemd`, a unidade `systemd.socket` pode ser usada de maneira semelhante. Nesta seção, usaremos o `xinetd` para interceptar conexões com o daemon `sshd` e iniciar este daemon no

momento adequado para demonstrar como o superdaemon era usado.

Antes de configurar o serviço xinetd, é necessária uma pequena preparação. Não importa se você usa um sistema baseado em Debian ou Red Hat. Embora estas explicações tenham sido testadas no Debian/GNU Linux 9.9, elas devem funcionar em qualquer sistema Linux atual com systemd sem nenhuma alteração significativa. Primeiro, verifique se os pacotes `openssh-server` e `xinetd` estão instalados. Em seguida, confira se o serviço SSH funciona com:

```
$ 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
```

Verifique também se o serviço SSH está escutando em sua porta de rede padrão 22:

```
# 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)
```

Finalmente, interrompa o serviço SSH com:

```
$ sudo systemctl stop sshd.service
```

No caso de você querer tornar esta mudança permanente, sobrevivendo à reinicialização, use `systemctl disable sshd.service`.

Agora você pode criar o arquivo de configuração do xinetd `/etc/xinetd.d/ssh` com algumas configurações básicas:

```
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
}

```

Reinic peace o serviço xinetd com:

```
$ sudo systemctl restart xinetd.service
```

Verifique nas conexões SSH de entrada qual serviço está escutando no momento.

```

$ 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)

```

Podemos ver que o serviço xinetd assumiu o controle do acesso à porta 22.

Eis mais alguns dados sobre a configuração do xinetd. O arquivo de configuração principal é `/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

```

Além das configurações padrão, há apenas uma diretiva para definir um diretório de inclusão. Neste diretório você pode definir um arquivo de configuração único para cada serviço que deseja que o xinetd gerencie. Fizemos isso acima para o serviço SSH e chamamos o arquivo de /etc/xinetd.d/ssh. O nome dos arquivos de configuração pode ser escolhido arbitrariamente, exceto por nomes de arquivos que contenham um ponto (.) ou que terminem com um til (~). Mas é comum atribuir ao arquivo o nome do serviço que se deseja configurar.

Alguns arquivos de configuração no diretório /etc/xinet.d/ já são fornecidos pela distribuição:

```
$ 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
```

Esses arquivos podem ser usados como modelos no raro caso de ser necessário usar alguns serviços legados, como o daytime, uma implementação ainda incipiente de um servidor de horário. Todos esses arquivos de modelo contêm a diretiva disable = yes.

Eis mais alguns detalhes sobre as diretivas usadas para ssh no arquivo de exemplo /etc/xinetd.d/ssh, acima.

```
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

Lista o serviço que o xinetd deve controlar. Podemos usar um número de porta, como 22, ou o nome mapeado para o número da porta em /etc/services, por exemplo ssh.

{

Abrimos uma chave para iniciar as configurações detalhadas.

disable

Para ativar essas configurações, defina esta diretiva como no. Se quiser desabilitar a configuração temporariamente, use yes.

socket_type

Escolha stream para os sockets TCP ou dgram para os sockets UDP e outros.

protocol

Escolha TCP ou UDP.

wait

Para as conexões TCP, esta diretiva costuma estar definida como no.

user

O serviço iniciado nesta linha será propriedade deste usuário.

server

Caminho completo para o serviço que deve ser iniciado por xinetd.

server_args

Aqui, podemos adicionar opções para o serviço. Quando iniciados por um super-servidor, muitos serviços requerem uma opção especial. Para o SSH, essa opção seria -i.

flags

Você pode escolher IPv4, IPv6 e outros.

interface

A interface de rede que o xinetd deve controlar. Nota: você também pode escolher a diretiva bind, que é simplesmente um sinônimo para interface.

{

Por fim, fechamos a chave.

Os sucessores dos serviços iniciados pelo super-servidor `xinetd` são as unidades de socket do `systemd`. Configurar essas unidades é muito simples, porque já existe uma unidade de socket do `systemd` predefinida para SSH. É preciso garantir que os serviços `xinetd` e SSH não estejam em execução.

Agora basta iniciar a unidade de socket SSH:

```
$ sudo systemctl start ssh.socket
```

Para verificar qual serviço está escutando agora na porta 22, usamos `lsof` novamente. Observe que a opção `-P` foi usada para mostrar o número da porta em vez do nome do serviço na saída:

```
$ 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)
```

Para completar esta sessão, tentamos fazer o login no servidor com um cliente SSH à escolha.

TIP Caso `systemctl start ssh.socket` não funcione com sua distribuição, experimente `systemctl start sshd.socket`.

Em busca de daemons desnecessários nos serviços

Por motivos de segurança, bem como para controlar os recursos do sistema, é importante ter uma visão geral dos serviços que estão em execução. Serviços desnecessários e não utilizados devem ser desativados. Por exemplo, caso você não precise distribuir páginas da web, não há necessidade de executar um servidor web como o Apache ou o nginx.

Nos sistemas baseados em Sys-V-init, podemos verificar o status de todos os serviços com o seguinte comando:

```
$ sudo service --status-all
```

Verifique se cada um dos serviços listados na saída do comando são necessários e desative todos os que forem desnecessários com (para sistemas baseados em Debian):

```
$ sudo update-rc.d SERVICE-NAME remove
```

Ou, nos sistemas baseados em Red Hat:

```
$ sudo chkconfig SERVICE-NAME off
```

Nos sistemas modernos baseados em systemd, podemos usar o seguinte comando para listar todos os serviços em execução:

```
$ systemctl list-units --state active --type service
```

Em seguida, desativamos as unidades de serviço desnecessárias com:

```
$ sudo systemctl disable UNIT --now
```

Este comando interrompe o serviço e o remove da lista de serviços, evitando assim que seja lançado na próxima inicialização do sistema.

Além disso, para obter um levantamento dos serviços de rede que estão à escuta, podemos usar netstat em sistemas mais antigos (desde que o pacote net-tools esteja instalado):

```
$ netstat -ltn
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:*
```

Ou, nos sistemas modernos, rodamos o comando equivalente ss (que significa “serviços de socket”):

```
$ 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
[::]:*

```

Usando TCP wrappers como uma espécie de firewall simples

Nos tempos em que não havia nenhum firewall disponível para Linux, TCP wrappers eram usados para proteger as conexões de rede em um host. Hoje em dia muitos programas não obedecem mais a TCP wrappers. Nas distribuições recentes baseadas em Red Hat (por exemplo, o Fedora 29), o suporte a TCP wrappers foi removido completamente. Porém, para oferecer suporte aos sistemas Linux legados que ainda usam TCP wrappers, vale a pena ter algum conhecimento básico sobre essa tecnologia específica.

Usaremos novamente o serviço SSH como exemplo básico. O serviço em nosso host de exemplo deve ser acessível apenas a partir da rede local. Primeiro, verificamos se o daemon SSH usa a biblioteca libwrap, que oferece suporte a TCP wrappers:

```
$ ldd /usr/sbin/sshd | grep "libwrap"
libwrap.so.0 => /lib/x86_64-linux-gnu/libwrap.so.0 (0x00007f91dbec0000)
```

A seguir, adicionamos a seguinte linha ao arquivo `/etc/hosts.deny`:

```
sshd: ALL
```

Finalmente, configuramos uma exceção no arquivo `/etc/hosts.allow` para conexões SSH da rede local:

```
sshd: LOCAL
```

As alterações entram em vigor imediatamente, sem a necessidade de reiniciar qualquer serviço. Para verificar, use o cliente `ssh`.

Exercícios Guiados

1. Como desbloquear a conta `emma`, anteriormente bloqueada?

2. Anteriormente, a conta `emma` tinha uma data de expiração definida. Como definir a data de expiração como `never` (nunca)?

3. Imagine que o serviço de impressão CUPS, que lida com trabalhos de impressão, não é necessário em seu servidor. Como desativar o serviço permanentemente? Como verificar se a porta apropriada não está mais ativa?

4. Você instalou o servidor web nginx. Como verificar se o nginx suporta TCP wrappers?

Exercícios Exploratórios

1. Verifique se a existência do arquivo `/etc/nologin` impede o login do usuário `root`.

2. A existência do arquivo `/etc/nologin` impede logins sem senha com chaves SSH?

3. O que acontece no login quando o arquivo `/etc/nologin` contém somente esta linha de texto:
`login currently is not possible?`

4. A usuária comum `emma` seria capaz de obter informações sobre o usuário `root` contidas no arquivo `/etc/passwd`, por exemplo com o comando `grep root /etc/passwd`?

5. A usuária comum `emma` seria capaz de recuperar informações sobre sua própria senha com hash contida no arquivo `/etc/shadow`, por exemplo com o comando `grep emma /etc/shadow`?

6. Quais etapas devem ser executadas para habilitar e verificar o antigo serviço `daytime` a ser gerenciado pelo `xinetd`? Observe que este é apenas um exercício exploratório, não faça isso em um ambiente de produção.

Resumo

Nesta lição, você aprendeu:

1. Em qual arquivo são armazenadas as senhas, bem como algumas configurações de segurança de senhas, por exemplo o tempo de expiração.
2. A finalidade do superdaemon `xinetd` e como colocá-lo em execução e iniciar o serviço `sshd` sob demanda.
3. Verificar quais serviços de rede estão em execução e desabilitar os serviços desnecessários.
4. Usar TCP wrappers como uma espécie de firewall simples.

Comandos usados na lição e nos exercícios:

`chage`

Altera a idade da senha de um usuário.

`chkconfig`

Um comando clássico inicialmente empregado nos sistemas baseados em Red Hat para definir se um serviço deve ser lançado no momento da inicialização ou não.

`netstat`

Um utilitário clássico (agora no pacote `net-tools`) que exibe os daemons que acessam as portas de rede do sistema e seu uso.

`nologin`

Um comando que pode ser usado no lugar do shell do usuário para impedir que ele faça login.

`passwd`

Usado para criar ou alterar a senha de um usuário.

`service`

Método antigo de controlar o status de um daemon, como interromper ou iniciar um serviço.

`ss`

O equivalente moderno de `netstat`, mas que também exibe mais informações sobre os diversos sockets em uso no sistema.

`systemctl`

Comando de controle do sistema, usado para controlar diversos aspectos dos serviços e sockets

em um computador baseado em systemd.

update-rc.d

Um comando clássico, semelhante ao `chkconfig`, que habilita ou desabilita o lançamento de um sistema no momento da inicialização nas distribuições baseadas em Debian.

xinetd

Um superdaemon capaz de controlar sob demanda o acesso a um serviço de rede, deixando assim o serviço inativo até que ele seja realmente chamado para realizar alguma tarefa.

Respostas aos Exercícios Guiados

- Como desbloquear a conta `emma`, anteriormente bloqueada?

O superusuário pode executar `passwd -u emma` para desbloquear a conta.

- Anteriormente, a conta `emma` tinha uma data de expiração definida. Como definir a data de expiração como `never` (nunca)?

O superusuário pode usar `chage -E -1 emma` para definir a data de expiração como nunca. Essa configuração pode ser conferida com `chage -l emma`.

- Imagine que o serviço de impressão CUPS, que lida com trabalhos de impressão, não é necessário em seu servidor. Como desativar o serviço permanentemente? Como verificar se a porta apropriada não está mais ativa?

Como superusuário, execute

```
systemctl disable cups.service --now
```

E para verificar

```
netstat -l | grep ":ipp" ` or `ss -l | grep ":ipp"
```

- Você instalou o servidor web nginx. Como verificar se o nginx suporta TCP wrappers?

O comando

```
ldd /usr/sbin/nginx | grep "libwrap"
```

mostra uma entrada, caso o nginx suporte TCP wrappers.

Respostas aos Exercícios Exploratórios

1. Verifique se a existência do arquivo `/etc/nologin` impede o login do usuário `root`.

O usuário `root` ainda é capaz de fazer login.

2. A existência do arquivo `/etc/nologin` impede logins sem senha com chaves SSH?

Sim, e os logins sem senha também são impedidos.

3. O que acontece no login quando o arquivo `/etc/nologin` contém somente esta linha de texto:
`login currently is not possible`?

A mensagem `login currently is not possible` será exibida e o login não será possível.

4. A usuária comum `emma` seria capaz de obter informações sobre o usuário `root` contidas no arquivo `/etc/passwd`, por exemplo com o comando `grep root /etc/passwd`?

Sim, pois todos os usuários têm permissão de leitura para esse arquivo.

5. A usuária comum `emma` seria capaz de recuperar informações sobre sua própria senha com hash contida no arquivo `/etc/shadow`, por exemplo com o comando `grep emma /etc/shadow`?

Não, pois os usuários comuns não têm permissão de leitura para esse arquivo.

6. Quais etapas devem ser executadas para habilitar e verificar o antigo serviço `daytime` a ser gerenciado pelo `xinetd`? Observe que este é apenas um exercício exploratório, não faça isso em um ambiente de produção.

Primeiro, altere o arquivo `/etc/xinetd.d/daytime` definindo a diretiva `disable = no`. Depois, reinicie o serviço `xinetd` com `systemctl restart xinetd.service` (ou `service xinetd restart` nos sistemas com SysV-Init). Confira se funcionou com `nc localhost daytime`. Em vez de `nc` também é possível usar `netcat`.



**Linux
Professional
Institute**

110.3 Proteção de dados com criptografia

Referência ao LPI objectivo

LPIC-1 version 5.0, Exam 102, Objective 110.3

Peso

4

Áreas chave de conhecimento

- Fazer uso e realizar a configuração básica do cliente OpenSSH 2.
- Entender a finalidade das chaves de servidor no OpenSSH 2.
- Configuração básica do GnuPG, seu uso e revogação.
- Usar o GPG para criptografar, descriptografar e verificar arquivos.
- Entender os túneis de porta do SSH (incluindo túneis X11).

Segue uma lista parcial dos arquivos, termos e utilitários utilizados

- `ssh`
- `ssh-keygen`
- `ssh-agent`
- `ssh-add`
- `~/.ssh/id_rsa` and `id_rsa.pub`
- `~/.ssh/id_dsa` and `id_dsa.pub`
- `~/.ssh/id_ecdsa` and `id_ecdsa.pub`
- `~/.ssh/id_ed25519` and `id_ed25519.pub`
- `/etc/ssh/ssh_host_rsa_key` and `ssh_host_rsa_key.pub`

- `/etc/ssh/ssh_host_dsa_key` and `ssh_host_dsa_key.pub`
- `/etc/ssh/ssh_host_ecdsa_key` and `ssh_host_ecdsa_key.pub`
- `/etc/ssh/ssh_host_ed25519_key` and `ssh_host_ed25519_key.pub`
- `~/.ssh/authorized_keys`
- `ssh_known_hosts`
- `gpg`
- `gpg-agent`
- `~/.gnupg/`



**Linux
Professional
Institute**

110.3 Lição 1

Certificação:	LPIC-1
Versão:	5.0
Tópico:	110 Segurança
Objetivo:	110.3 Proteção de dados com criptografia
Lição:	1 de 2

Introdução

A proteção de dados com criptografia é uma atribuição de extrema importância na administração de sistemas—ainda mais quando se trata de acessar sistemas remotamente. Ao contrário de soluções inseguras como *telnet*, *rlogin* ou *FTP*, o protocolo *SSH* (*Secure Shell*) foi projetado tendo em mente a segurança. Graças à criptografia de chave pública, ele autentica hosts e usuários e criptografa todas as trocas de informações subsequentes. Além disso, o *SSH* pode ser usado para estabelecer *túneis de porta*, que—entre outras coisas—permite que um protocolo não criptografado transmita dados por meio de uma conexão *SSH* criptografada. A versão atual recomendada do protocolo *SSH* é a 2.0. O *OpenSSH* é uma implementação gratuita e de código aberto do protocolo *SSH*.

Esta lição cobrirá a configuração básica do cliente *OpenSSH*, bem como a função das chaves de host do servidor *OpenSSH*. O conceito de túneis de porta *SSH* também será discutido. Usaremos duas máquinas com a seguinte configuração:

Função da máquina	SO	Endereço IP	Nome do host	Usuário
Cliente	Debian GNU/Linux 10 (buster)	192.168.1.55	debian	carol
Servidor	openSUSE Leap 15.1	192.168.1.77	halof	ina

Configuração e uso básico do cliente OpenSSH

Embora o servidor e o cliente OpenSSH venham em pacotes separados, normalmente é possível instalar um metapacote que fornece ambos ao mesmo tempo. Para estabelecer uma sessão remota com o servidor SSH, usamos o comando `ssh`, especificando o usuário que desejamos conectar na máquina remota e o endereço IP ou nome de host da máquina remota. Na primeira vez que nos conectamos a um host remoto, recebemos uma mensagem como esta:

```
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 SHA256:5JF7anupYipByCQm2BPvDHRVFJJixeslmppi2NwATYI.
Are you sure you want to continue connecting (yes/no)?
```

Após digitar `yes` e dar Enter, será solicitada a senha do usuário remoto. Depois disso aparece uma mensagem de aviso e, em seguida, é feita a conexão ao host remoto:

```
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:~>
```

As mensagens são autoexplicativas: como é a primeira vez que você estabeleceu uma conexão com o servidor remoto 192.168.1.77, sua autenticidade não pode ser verificada em nenhum banco de dados. Assim, o servidor remoto forneceu uma ECDSA key fingerprint (impressão digital) de sua chave pública (usando a função hash SHA256). Depois que você aceitou a conexão, a chave pública do servidor remoto foi adicionada ao banco de dados de *hosts conhecidos*, permitindo assim a autenticação do servidor para conexões futuras. Esta lista de chaves públicas de *hosts conhecidos* é mantida no arquivo `known_hosts`, que reside em `~/.ssh`:

```
ina@halof:~> exit
logout
Connection to 192.168.1.77 closed.
carol@debian:~$ ls .ssh/
known_hosts
```

Tanto `.ssh` quanto `known_hosts` foram criados após a primeira conexão remota ter sido estabelecida. `~/.ssh` é o diretório padrão para configurações e informações de autenticação específicas do usuário.

NOTE Também é possível usar `ssh` para executar um único comando no host remoto e em seguida retornar ao terminal local (p. ex. executando `ssh ina@halof ls`).

Se você estiver usando o mesmo usuário nos hosts local e remoto, não há necessidade de especificar o nome de usuário ao estabelecer a conexão SSH. Por exemplo, se você estiver logado como a usuária `carol` no `debian` e quiser se conectar ao `halof` também como usuária `carol`, bastaria digitar `ssh 192.168.1.77` ou `ssh halof` (se o nome puder ser resolvido):

```
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:~>
```

Agora, suponha que você queira estabelecer uma nova conexão remota com um host que por acaso tem o mesmo endereço IP de `halof` (o que é comum se você usa DHCP em sua LAN). Aparecerá um aviso sobre a possibilidade de um ataque *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:KH4q3vP6C7e0SEjyG8Wlz9fVlf+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.
```

Como não se trata de um ataque *man-in-the-middle*, você pode adicionar com segurança a impressão digital de chave pública do novo host a `.ssh/known_hosts`. Como a mensagem indica, o comando `ssh-keygen -f "/home/carol/.ssh/known_hosts" -R "192.168.1.77"` pode ser usado para remover a chave *ofensiva* (outra alternativa é usar `ssh-keygen -R 192.168.1.77` para remover todas as chaves pertencentes a 192.168.1.77 de `~/ssh/known_hosts`). Depois disso, será possível estabelecer uma conexão ao novo host.

Logins baseados em chave

Podemos configurar o cliente SSH para não fornecer nenhuma senha no login, e sim usar chaves públicas. Este é o método mais usado para a conexão a um servidor remoto via SSH, pois é muito mais seguro. A primeira coisa a fazer é criar um par de chaves na máquina cliente. Para isso, usamos `ssh-keygen` com a opção `-t`, especificando o tipo de criptografia desejado (Em nosso caso, *Elliptic Curve Digital Signature Algorithm*, ou Algoritmo de Assinatura Digital de Curvas Elípticas). Em seguida, precisamos indicar o caminho para salvar o par de chaves (`~/ssh/` é conveniente, assim como o local padrão), além de uma frase secreta. Embora a frase secreta seja opcional, é altamente recomendável usar sempre uma.

```
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 & O  |
|   @ ^ =  |
|   *.* @.  |
|   o.o+B+o |
```

+---- [SHA256] ----+

NOTE

Ao criar o par de chaves, podemos passar a opção `-b` para `ssh-keygen` para especificar o tamanho da chave em bits (p. ex.: `ssh-keygen -t ecdsa -b 521`).

O comando anterior mostra mais dois arquivos em seu diretório `~/.ssh`:

```
carol@debian:~/ssh$ ls
id_ecdsa  id_ecdsa.pub  known_hosts
```

id_ecdsa

Esta é a sua chave privada.

id_ecdsa.pub

Esta é a sua chave pública.

NOTE

Na criptografia assimétrica (também conhecida como criptografia de chave pública), as chaves pública e privada estão matematicamente relacionadas uma com a outra de tal forma que tudo o que é criptografado por uma só pode ser descriptografado pela outra.

A seguir, você precisa adicionar sua chave pública ao arquivo `~/.ssh/authorized_keys` do usuário que vai se logar no host remoto (se o diretório `~/.ssh` ainda não existir, é preciso criá-lo primeiro). Há várias maneiras de copiar sua chave pública para o servidor remoto: usar uma unidade flash USB, através do comando `scp`—que transfere o arquivo usando SSH—ou ler (com `cat`) o conteúdo de sua chave pública e canalizá-lo para `ssh` desta maneira:

```
carol@debian:~/ssh$ cat id_ecdsa.pub | ssh ina@192.168.1.77 'cat >> .ssh/authorized_keys'
Password:
```

Uma vez que sua chave pública tenha sido adicionada ao arquivo `authorized_keys` no host remoto, podem ocorrer dois cenários ao se tentar estabelecer uma nova conexão:

- Se você não tiver fornecido uma frase secreta ao criar o par de chaves, será conectado automaticamente. Embora conveniente, este método pode ser inseguro dependendo da situação:

```
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:~>
```

- Se você forneceu uma frase secreta ao criar o par de chaves, terá de inseri-la em todas as conexões, como se fosse uma senha. Além da chave pública, este método adiciona uma camada extra de segurança na forma de uma frase secreta e pode — portanto — ser considerado mais seguro. No que diz respeito à conveniência, no entanto, é exatamente o mesmo que digitar uma senha sempre que se estabelecer uma conexão. Se você não usar uma frase secreta e alguém conseguir obter seu arquivo de chave SSH privada, essa pessoa terá acesso a todos os servidores em que sua chave pública estiver instalada.

```
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:~>
```

Porém, existe uma maneira que combina segurança e conveniência: usar o *agente de autenticação SSH* (`ssh-agent`). O agente de autenticação precisa gerar seu próprio shell e guarda na memória suas chaves privadas — para autenticação de chave pública — pelo restante da sessão. Vamos ver como ele funciona com um pouco mais de detalhes:

1. Use `ssh-agent` para iniciar um novo shell Bash:

```
carol@debian:~/ssh$ ssh-agent /bin/bash
carol@debian:~/ssh$
```

2. Use o comando `ssh-add` para adicionar sua chave privada a uma área segura da memória. Se tiver fornecido uma frase secreta ao gerar o par de chaves — o que é recomendado por razões de segurança — esse será o momento de inseri-la:

```
carol@debian:~/ssh$ ssh-add
Enter passphrase for /home/carol/.ssh/id_ecdsa:
Identity added: /home/carol/.ssh/id_ecdsa (carol@debian)
```

Depois de adicionar sua identidade, você pode fazer o login em qualquer servidor remoto no qual sua chave pública esteja presente sem precisar digitar sua senha novamente. Nos sistemas modernos, é comum executar este comando ao inicializar o computador, pois ele permanecerá na memória até que a máquina seja desligada (ou a chave seja descarregada manualmente).

Vamos terminar esta seção listando os quatro tipos de algoritmos de chave pública que podem ser especificados com `ssh-keygen`:

RSA

O nome foi dado em homenagem aos criadores Ron Rivest, Adi Shamir e Leonard Adleman. Foi publicado em 1977. É considerado seguro e ainda é muito usado hoje em dia. Seu tamanho mínimo de chave é de 1024 bits (o padrão é 2048).

DSA

O *Digital Signature Algorithm* é comprovadamente inseguro e passou a ser preterido a partir do OpenSSH 7.0. As chaves DSA devem ter exatamente 1024 bits.

ecdsa

O *Elliptic Curve Digital Signature Algorithm* é um aprimoramento do DSA e—como tal—considerado mais seguro. Ele emprega a criptografia de curvas elípticas. O comprimento da chave ECDSA é determinado por um dos três tamanhos possíveis de curvas elípticas em bits: 256, 384 ou 521.

ed25519

É uma implementação do EdDSA—*Edwards-curve Digital Signature Algorithm*—que usa a curva 25519, mais forte. É considerado o mais seguro de todos. Todas as chaves Ed25519 têm um comprimento fixo de 256 bits.

NOTE

Quando invocado sem a especificação `-t`, o `ssh-keygen` gera um par de chaves RSA por padrão.

O papel das chaves de host do servidor OpenSSH

O diretório de configuração global do OpenSSH reside no diretório `/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

Além de `moduli` e dos arquivos de configuração do cliente (`ssh_config`) e do servidor (`sshd_config`), quatro pares de chaves—um par de chaves para cada algoritmo suportado—são criados quando o servidor *OpenSSH* é instalado. Como já dissemos, o servidor usa essas *chaves de host* para se identificar junto aos clientes conforme necessário. Seu padrão de nome é o seguinte:

Chaves privadas

`ssh_host_prefixo + algoritmo + sufixo key` (p. ex.: `ssh_host_rsa_key`)

Chaves públicas (ou impressões digitais de chave pública)

`ssh_host_prefixo + algoritmo + sufixo key .pub` (p. ex.: `ssh_host_rsa_key .pub`)

NOTE

Uma impressão digital é criada aplicando-se uma função hash criptográfica a uma chave pública. Como as impressões digitais são mais curtas do que as chaves a que se referem, elas são úteis para simplificar certas tarefas de gerenciamento de chaves.

As permissões nos arquivos que contêm as chaves privadas são `0600` ou `-rw-----`: legíveis e graváveis somente pelo proprietário (root). Por outro lado, todos os arquivos de chave pública também são legíveis por membros do grupo do proprietário e por todos os outros (`0644` ou `-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

```

Para ver as impressões digitais das chaves, passe a opção `-l` para `ssh-keygen`. Também é preciso usar `-f` para especificar o caminho do arquivo da chave:

```

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+1JfZ4WBRZhd3rD0sc2zlA root@halof (ED25519)
```

Para visualizar a impressão digital da chave, bem como sua arte aleatória (randomart), basta adicionar a opção `-v`:

```
halof:~ # ssh-keygen -lv -f /etc/ssh/ssh_host_ed25519_key.pub
256 SHA256:8cnPrinC49ZHc+/9Ai5pV+1JfZ4WBRZhd3rD0sc2zlA root@halof (ED25519)
+-- [ED25519 256] --+
|          +oo|
|          .+o.|
|          .   ..E.|
|          + .  +.o|
|          S +  *o|
|          ooo 0o=|
|          . . . =o+.==|
|          = o =oo o=o|
|          o.o +o+..o.+|
+---[SHA256]---
```

Túneis de porta SSH

O OpenSSH apresenta um recurso de encaminhamento muito poderoso, por meio do qual o tráfego em uma porta de origem é tunelado — e criptografado — por meio de um processo SSH, que em seguida o redireciona para uma porta em um host de destino. Esse mecanismo é chamado de *tunelamento de porta* (port tunnelling) ou *encaminhamento de porta* (port forwarding) e oferece vantagens importantes, como as seguintes:

- Permite contornar firewalls para acessar portas em hosts remotos.
- Possibilita o acesso externo a um host em sua rede privada.
- Fornece criptografia para todas as trocas de dados.

Grosso modo, podemos diferenciar entre o tunelamento de portas local e remoto.

Túnel de porta local

Definimos uma porta localmente para encaminhar o tráfego para o host de destino por meio do processo SSH que fica entre os dois. O processo SSH pode ser executado no host local ou em um servidor remoto. Por exemplo, se por algum motivo você quisesse fazer um túnel de conexão para www.gnu.org por SSH usando a porta 8585 em sua máquina local, a operação seria semelhante à

seguinte:

```
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
```

Eis a explicação: com a opção `-L`, especificamos que a porta local 8585 deve se conectar à porta `http 80` em `www.gnu.org` usando o processo SSH executado em `debian`—nossa *host local*. Poderíamos ter escrito `ssh -L 8585:www.gnu.org:80 localhost` com o mesmo efeito. Se agora visitássemos `http://localhost:8585` com um navegador web, seríamos encaminhados a `www.gnu.org`. Para fins de demonstração, usaremos o `lynx` (o navegador web clássico em modo texto):

```
carol@debian:~$ lynx http://localhost:8585
(...)
* Back to Savannah Homepage
  * Not Logged in
  * Login
  * New User
  * This Page
  * Language
  * Clean Reload
  * Printer Version
  * Search
  *
(...)
```

Se você quisesse fazer exatamente a mesma coisa, mas se conectando por meio de um processo SSH em execução no halof, o procedimento seria o seguinte:

```
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
*
(....)
```

É importante observar três detalhes no comando:

- Graças à opção `-N`, não fizemos login no `halof`, mas sim o encaminhamento de porta.
- A opção `-f` disse ao SSH para rodar em segundo plano.
- Especificamos o usuário `ina` para o encaminhamento: `ina@192.168.1.77`

Túnel de porta remota

No túnel de porta remota (ou encaminhamento de porta reversa), o tráfego vindo de uma porta no servidor remoto é encaminhado para o processo SSH em execução em seu host local e, de lá, para a porta especificada no servidor de destino (que também pode ser a sua máquina local). Por exemplo, digamos que você queira permitir que alguém de fora de sua rede acesse o servidor web Apache em execução no seu host local através da porta 8585 do servidor SSH rodando em `halof` (192.168.1.77). Você continuaria com o seguinte comando:

```
carol@debian:~$ ssh -R 8585:localhost:80 -Nf ina@192.168.1.77
Enter passphrase for key '/home/carol/.ssh/id_ecdsa':
carol@debian:~$
```

Agora, qualquer pessoa que estabelecer uma conexão com o `halof` na porta 8585 verá a página inicial padrão do Apache2 do 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

Existe um terceiro tipo de encaminhamento de porta mais complexo, que está fora do escopo desta lição: o *encaminhamento de porta dinâmico*. Em vez de interagir com uma única porta, esse tipo de encaminhamento usa diversas comunicações TCP em uma variedade de portas.

Túneis X11

Agora que você já conhece os túneis de portas, vamos encerrar esta lição discutindo o tunelamento X11 (também conhecido como *X11forwarding*). Por meio de um túnel X11, o *X Window System* no host remoto é encaminhado para sua máquina local. Para isso, basta passar a opção `-X` ao `ssh`:

```
carol@debian:~$ ssh -X ina@halof
...
```

Agora você pode iniciar um aplicativo gráfico, como o navegador `firefox`, com o seguinte resultado: o aplicativo será executado no servidor remoto, mas sua exibição será encaminhada ao seu host local.

Se iniciarmos uma nova sessão SSH com a opção `-x`, *X11forwarding* será desabilitado. Tente iniciar o `firefox` agora e aparecerá um erro semelhante ao seguinte:

```
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

As três diretivas de configuração relacionadas ao encaminhamento de porta local, encaminhamento de porta remota e encaminhamento de X11 são

`AllowTcpForwarding`, `GatewayPorts` e `X11Forwarding`, respectivamente. Para maiores informações, digite `man ssh_config` e/ou `man sshd_config`.

Exercícios Guiados

1. Conectado como a usuária `sonya` em sua máquina cliente, execute as seguintes tarefas de SSH no servidor remoto `halof`:

- Execute o comando para listar o conteúdo de `~/.ssh` como a usuária `serena` no hospedeiro remoto; em seguida, volte ao seu terminal local.

- Faça login como a usuária `serena` no host remoto.

- Faça login como a usuária `sonya` no host remoto.

- Exclua todas as chaves pertencentes a `halof` de seu arquivo local `~/.ssh/known_hosts`.

- Em sua máquina cliente, crie um par de chaves `ecdsa` de 256 bits.

- Em sua máquina cliente, crie um par de chaves `ed25519` de 256 bits.

2. Coloque as seguintes etapas na ordem certa para estabelecer uma conexão SSH usando o *agente de autenticação SSH*:

- No cliente, inicie um novo shell Bash para o *agente de autenticação* com `ssh-agent /bin/bash`.
- No cliente, crie um par de chaves usando `ssh-keygen`.
- No cliente, adicione sua chave privada a uma área segura da memória com `ssh-add`.
- Adicione a chave pública do seu cliente ao arquivo `~/.ssh/authorized_keys` do usuário com o qual será feito o login no host remoto.
- Se ainda não existir, crie `~/.ssh` para o usuário com o qual será feito o login no servidor.
- Conecte-se ao servidor remoto.

A ordem correta é:

Passo 1:	
-----------------	--

Passo 2:	
Passo 3:	
Passo 4:	
Passo 5:	
Passo 6:	

3. Com relação ao *encaminhamento de portas*, qual opção e diretiva é usada para os seguintes tipos de túneis:

Tipo de túnel	Opção	Diretiva
Local		
Remoto ou Reverso		
X		

4. Suponha que você digita o comando `ssh -L 8888:localhost:80 -Nf ina@halof` no terminal de sua máquina cliente. Ainda nessa máquina, você aponta um navegador para `http://localhost:8888`. O que aparece?

Exercícios Exploratórios

1. Com relação às diretivas de segurança do SSH:

- Qual diretiva é usada em `/etc/ssh/sshd_config` para permitir logins de `root`:

- Qual diretiva seria usada em `/etc/ssh/sshd_config` para especificar que somente uma conta local pode aceitar conexões SSH:

2. Ao definir o mesmo usuário no cliente e no servidor, qual comando `ssh` podemos usar para transferir a chave pública do cliente para o servidor, sendo possível fazer o login através da autenticação de chave pública?

3. Crie dois túneis de portas locais em um único comando, encaminhando as portas locais sem privilégios 8080 e 8585, através do servidor remoto `halof`, para os websites `www.gnu.org` e `www.melpa.org`, respectivamente. Use o usuário `ina` no servidor remoto e não se esqueça das opções `-Nf`:

Resumo

Nesta lição, discutimos o *OpenSSH* 2, que usa o protocolo *Secure Shell* para criptografar as comunicações entre o servidor e o cliente. Você aprendeu:

- como fazer login em um servidor remoto.
- como executar comandos remotamente.
- como criar pares de chaves.
- como estabelecer logins baseados em chaves.
- como usar o *agente de autenticação* para garantir maior segurança e conveniência.
- os algoritmos de chave pública suportados pelo *OpenSSH*: RSA, DSA, ecdsa, ed25519.
- a função das chaves de host *OpenSSH*.
- como criar túneis de portas: local, remoto e X.

Os seguintes comandos foram discutidos nesta lição:

ssh

Fazer login ou executar comandos em uma máquina remota

ssh-keygen

Gerar, gerenciar e converter chaves de autenticação.

ssh-agent

Agente de autenticação do OpenSSH.

ssh-add

Adiciona identidades de chave privada ao agente de autenticação.

Respostas aos Exercícios Guiados

1. Conectado como a usuária `sonya` em sua máquina cliente, execute as seguintes tarefas de SSH no servidor remoto `halof`:

- Execute o comando para listar o conteúdo de `~/.ssh` como a usuária `serena` no hospedeiro remoto; em seguida, volte ao seu terminal local.

```
ssh serena@halof ls .ssh
```

- Faça login como a usuária `serena` no host remoto.

```
ssh serena@halof
```

- Faça login como a usuária `sonya` no host remoto.

```
ssh halof
```

- Exclua todas as chaves pertencentes a `halof` de seu arquivo local `~/.ssh/known_hosts`.

```
ssh-keygen -R halof
```

- Em sua máquina cliente, crie um par de chaves `ecdsa` de 256 bits.

```
ssh-keygen -t ecdsa -b 256
```

- Em sua máquina cliente, crie um par de chaves `ed25519` de 256 bits.

```
ssh-keygen -t ed25519
```

2. Coloque as seguintes etapas na ordem certa para estabelecer uma conexão SSH usando o *agente de autenticação SSH*:

- No cliente, inicie um novo shell Bash para o *agente de autenticação* com `ssh-agent /bin/bash`.
- No cliente, crie um par de chaves usando `ssh-keygen`.
- No cliente, adicione sua chave privada a uma área segura da memória com `ssh-add`.

- Adicione a chave pública do seu cliente ao arquivo `~/.ssh/authorized_keys` do usuário com o qual será feito o login no host remoto.
- Se ainda não existir, crie `~/.ssh` para o usuário com o qual será feito o login no servidor.
- Conecte-se ao servidor remoto.

A ordem correta é:

Passo 1:	No cliente, crie um par de chaves usando <code>ssh-keygen</code> .
Passo 2:	Se ainda não existir, crie <code>~/.ssh</code> para o usuário com o qual será feito o login no servidor.
Passo 3:	Adicione a chave pública do seu cliente ao arquivo <code>~/.ssh/authorized_keys</code> do usuário com o qual será feito o login no host remoto.
Passo 4:	No cliente, inicie um novo shell Bash para o <i>agente de autenticação</i> com <code>ssh-agent /bin/bash</code> .
Passo 5:	No cliente, adicione sua chave privada a uma área segura da memória com <code>ssh-add</code> .
Passo 6:	Conecte-se ao servidor remoto.

3. Com relação ao *encaminhamento de portas*, qual opção e diretiva é usada para os seguintes tipos de túneis:

Tipo de túnel	Opção	Diretiva
Local	<code>-L</code>	<code>AllowTcpForwarding</code>
Remoto ou Reverso	<code>-R</code>	<code>GatewayPorts</code>
X	<code>-X</code>	<code>X11Forwarding</code>

4. Suponha que você digita o comando `ssh -L 8888:localhost:80 -Nf ina@halof` no terminal de sua máquina cliente. Ainda nessa máquina, você aponta um navegador para `http://localhost:8888`. O que aparece?

A página inicial de `halof` do servidor web, que é como `localhost` é entendido da perspectiva do servidor.

Respostas aos Exercícios Exploratórios

1. Com relação às diretivas de segurança do SSH:

- Qual diretiva é usada em /etc/ssh/sshd_config para permitir logins de root:

PermitRootLogin

- Qual diretiva seria usada em /etc/ssh/sshd_config para especificar que somente uma conta local pode aceitar conexões SSH:

AllowUsers

2. Ao definir o mesmo usuário no cliente e no servidor, qual comando ssh podemos usar para transferir a chave pública do cliente para o servidor, sendo possível fazer o login através da autenticação de chave pública?

ssh-copy-id

3. Crie dois túneis de portas locais em um único comando, encaminhando as portas locais sem privilégios 8080 e 8585, através do servidor remoto halof, para os websites www.gnu.org e www.melpa.org, respectivamente. Use o usuário ina no servidor remoto e não se esqueça das opções -Nf:

ssh -L 8080:www.gnu.org:80 -L 8585:www.melpa.org:80 -Nf ina@halof



**Linux
Professional
Institute**

110.3 Lição 2

Certificação:	LPIC-1
Versão:	5.0
Tópico:	110 Segurança
Objetivo:	110.3 Proteção de dados com criptografia
Lição:	2 de 2

Introdução

Na lição anterior, aprendemos como usar o *OpenSSH* para criptografar sessões de login remoto, bem como qualquer outra troca de informações subsequente. Claro, existem outras situações em que é necessário criptografar arquivos ou emails para que cheguem ao destinatário em segurança, longe de olhares indiscretos. Também pode ser preciso assinar digitalmente esses arquivos ou mensagens para evitar que sejam adulterados.

Para essas finalidades, recomendamos vivamente o *GNU Privacy Guard* (também conhecido como *GnuPG* ou simplesmente *GPG*), uma implementação gratuita e de código aberto do *Pretty Good Privacy (PGP)*, que é proprietário. O *GPG* usa o padrão *OpenPGP*, definido pelo *Grupo de Trabalho OpenPGP* da *Internet Engineering Task Force (IETF)* no RFC 4880. Nesta lição, abordaremos os fundamentos do *GNU Privacy Guard*.

Como executar a configuração, uso e revogação básicos do GnuPG

Como no caso do SSH, o mecanismo que rege o GPG é a *criptografia assimétrica* ou *criptografia de*

chave pública. Um usuário gera um par de chaves composto de uma *chave privada* e uma *chave pública*. As chaves estão matematicamente relacionadas de tal forma que o que é criptografado por uma só pode ser descriptografado pela outra. Para que a comunicação ocorra com sucesso, o usuário deve enviar sua chave pública ao destinatário pretendido.

Configuração e uso do GnuPG

O comando para trabalhar com o GPG é `gpg`. Ele inclui diversas opções para realizar diferentes tarefas. Vamos começar gerando um par de chaves como a usuária `carol`. Para isso, usamos o comando `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:
(....)
```

Depois de informar—entre outras coisas—que o diretório de configuração `~/.gnupg` e seu chaveiro público `~/.gnupg/pubring.kbx` foram criados, o `gpg` pede seu nome real e endereço de 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?
```

Se você estiver de acordo com o USER-ID resultante e pressionar `o`, será solicitada uma frase secreta (recomenda-se escolher uma senha complexa):

```
| Please enter the passphrase to
| protect your new key
```

```
| Passphrase: |
```

```
(...)
```

Aparecem mais algumas mensagens sobre a criação de outros arquivos, bem como as próprias chaves, e o processo de geração de chaves é concluído:

```
(...)
```

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]
```

Agora você pode ver o que está dentro do diretório `~/.gnupg` (o diretório de configuração do GPG):

```
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
```

Vamos explicar o uso de cada arquivo:

openpgp-revocs.d

O certificado de revogação criado junto com o par de chaves é mantido aqui. As permissões deste diretório são bastante restritivas, pois qualquer pessoa que tenha acesso ao certificado pode revogar a chave (trataremos da revogação da chave na próxima subseção).

private-keys-v1.d

Este é o diretório que preserva suas chaves privadas, portanto as permissões são restritivas.

pubring.kbx

Este é o seu chaveiro público. Ele armazena suas próprias chaves públicas, bem como quaisquer outras que tenham sido importadas.

trustdb.gpg

Banco de dados de confiabilidade. Está relacionado ao conceito de *Web of Trust* (que está fora do escopo desta lição).

NOTE A chegada do *GnuPG 2.1* trouxe algumas mudanças importantes, como o desaparecimento dos arquivos `secring.gpg` e `pubring.gpg` em favor de `private-keys-v1.d` e `pubring.kbx`, respectivamente.

Depois de criar o par de chaves, você pode visualizar suas chaves públicas com `gpg --list-keys` — que exibe o conteúdo de seu chaveiro público:

```
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]
```

A string hexadecimal `D18FA0021F644CDAF57FD0F919BBEFD16813034E` é sua *impressão digital da chave pública*.

NOTE Além de `USER-ID` (carol, no exemplo), também existe o `KEY-ID`. O `KEY-ID` consiste nos últimos 8 dígitos hexadecimais da impressão digital da chave pública (6813 034E). Para conferir a impressão digital de sua chave, use o comando `gpg --fingerprint USER-ID`.

Distribuição e revogação de chaves

Agora que você tem sua chave pública, é preciso salvá-la (ou seja, *exportá-la*) em um arquivo para ser possível disponibilizá-la para os futuros destinatários. Eles a usarão para criptografar arquivos ou mensagens destinadas a você (como você é o único detentor da chave privada, também será o único capaz de descriptografar e ler esses arquivos). Da mesma forma, seus destinatários também a usarão para descriptografar e verificar suas mensagens/arquivos criptografados ou assinados. O comando a ser usado é `gpg --export`, seguido pelo `USER-ID` e um redirecionamento para o nome do arquivo de saída de sua escolha:

```
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

Passe as opções `-a` ou `--armor` para `gpg --export` (p. ex.: `gpg --export --armor carol > carol.pub.key`) para criar uma saída ASCII blindada (ao invés do formato OpenPGP binário padrão) que pode ser enviada com segurança por email.

Como já observado, você deve enviar seu arquivo de chave pública (`carol.pub.key`) para o destinatário com quem deseja trocar informações. Por exemplo, vamos enviar o arquivo de chave pública para `ina`, no servidor remoto `halof`, usando `scp` (*cópia segura*):

```
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` agora está em posse de `carol.pub.key`. Ela a usará para criptografar um arquivo e enviá-lo para `carol` na próxima seção.

NOTE

Outra maneira de distribuir chaves públicas é usar *servidores de chave*: você envia sua chave pública para o servidor com o comando `gpg --keyserver keyserver-name --send-keys KEY-ID` e os outros usuários podem baixá-la (ou seja, importá-la) com `gpg --keyserver keyserver-name --recv-keys KEY-ID`.

Para encerrar esta seção, vamos falar da revogação da chave. A revogação da chave deve ser usada quando suas chaves privadas forem comprometidas ou aposentadas. O primeiro passo é criar um certificado de revogação passando ao `gpg` a opção `--gen-revoke` seguida pelo `USER-ID`.

Para salvar o certificado resultante em um arquivo (em vez de imprimi-lo na tela do terminal), inclua a opção `--output`, seguida pelo nome de um arquivo de destino, antes de `--gen-revoke`. As mensagens de saída ao longo do processo de revogação são bastante autoexplicativas:

```
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!
```

O certificado de revogação foi salvo no arquivo `revocation_file.asc` (`asc` indica o formato 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
```

```
bGFwdG9wAAoJEJ5XDesiwnhxT9YMAKkjQiMpo9Uyi9hyvukPPSrlCmtAGLk4pKS
pLZfzA5kxa+HPQwBglAEvfNRR6VMxqXUgUGYC/IAyQQM62oNAcY2PCPrxyJNgVF7
814mMZKvW++5ikjZwyg6WWV0+w6oroeo9qrufJfcu752p4T+9gsHVa2r+KRqcPQe
aZ65sAvsBJlcsUDZqfWUXg2kQp9mNPCdQuqvDaKRgNCHA1zbzNFzXWVd2X5RgFo5
nY+tUP8ZQA9DTQPBLPcggiCmfLopMPZYB2bft5geb2mMi2oNpf9CNPdQkdccimNV
aRjqdUP9C89PwTafBQkQiONlsR/dWTFcqprG5K0WQPA7xjeMV8wretdEgsyTxqHp
v1iRzwjshiJCKBXXvz7wSmQrJ40fiMDHeS4ipR0AYd08QCzm0zmcFQKikGSHGMy1
z/YRltd6NZIKjf1TD0nTrFnRvPdsZ01KYSArbfqNrHRBQkgir0D4JPI1tYKTffq
i0eZFx25K+fj2+0AJjvrbe4HD05m+Q==

=umI8
-----END PGP PUBLIC KEY BLOCK-----
```

Para revogar de fato sua chave privada, agora você precisa mesclar o certificado com a chave, o que é feito importando o arquivo do certificado de revogação para o seu chaveiro:

```
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
```

A seguir, liste suas chaves, e você será informado sobre a chave revogada:

```
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>
```

Para terminar, não deixe de disponibilizar a chave revogada para qualquer terceiro que tenha chaves públicas associadas a ela (incluindo servidores de chaves).

Como usar o GPG para criptografar, descriptografar, assinar e verificar arquivos

Na seção anterior, carol enviou sua chave pública para ina. Vamos usá-la agora para aprender como o GPG é capaz de criptografar, descriptografar, assinar e verificar arquivos.

Criptografando e descriptografando arquivos

Primeiro, ina deve importar a chave pública de carol (carol.pub.key) para seu chaveiro antes de poder começar a trabalhar com ela:

```
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]
```

Em seguida, criamos um arquivo com um texto qualquer e o criptografamos usando gpg (como você não assinou a chave de carol, o programa pergunta explicitamente se deseja usar essa chave):

```
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

Vamos analisar o comando gpg:

--output encrypted-message

Especificação do nome de arquivo para a versão criptografada do arquivo original (encrypted-message no exemplo).

--recipient carol

Especificação do USER-ID do destinatário (carol em nosso exemplo). Se não for fornecido, o GnuPG irá solicitá-lo (a menos que **--default-receiver** esteja especificado).

--armor

Esta opção produz uma saída ASCII blindada, que pode ser copiada para um email.

--encrypt unencrypted-message

Especificação do nome do arquivo original a ser criptografado.

Em seguida, enviamos **encrypted-message** para **carol** em **debian** usando **scp**:

```
ina@halof:~> scp encrypted-message carol@debian:/home/carol/
carol@debian's password:
encrypted-message                                         100%   736
1.8MB/s   00:00
```

Se você se logar como **carol** agora e tentar ler **encrypted-message**, poderá constatar que ela está realmente criptografada e — portanto — ilegível:

```
carol@debian:~$ cat encrypted-message
-----BEGIN PGP MESSAGE-----
hQGMAwInNHzJKlyxAQv/bkJ8Ubs/xya35sbv6kdRKm1C70NLxL30ueWA4mCs0Y/P
GBna6ZEUCrMEgl/rCyByj3Yq74kuiTmxzAIRUDdvHfj0Ttr0WjVAqIn/fPSfMkj
dTxDKo1i55tLJ+sj17dGMZDcNBinBTP4U1atuN71A5w7vH+XpcEsRcFQLKiS0mYTt
F7SN3/5x5J6io4ISn+b0KbjgiJNNx+Ne/ub4Uzk4N1K7tmBklyC1VRualtxcG7R9
1klBPYSld6fTdDwT1Y4MofpyILAiGMZvUR1RXauEKf70IzwC5gWU+UQPSgeCdKQu
X7QL0ZIBS0Ug2XKr01k93lmDjf8PWsRIml6n/hNela0BA3HMP0b60zv1gFeEsFvC
IxhUYPb+rFuNFTMEB7xIO94AAmWB9N4qknMxdDqNE8WhA728Plw6y8L2ngsplY15
MR4lIFDpljA/CcVh4BXVe9j0TdFWDUkrFMfaIfcPQwKLXEYJp19XYIaaEazkOs5D
W4pENN0Y0cX0KWyAYX6r018BF0rq/HMenQwqAVXMG3s8ATuU0eqjBbR1x1qCvRQP
CR/3V73aQwc2j5ioQmhWYpqxiro0yKX2Ar/E6rzYjtJYrq+CUk803JoBaudknNFj
pwuRwF1amwnSZ/MZ/9kMKQ==
=g1jw
-----END PGP MESSAGE-----
```

No entanto, como você possui a chave privada, pode descriptografar a mensagem facilmente passando ao **gpg** a opção **--decrypt** seguida pelo caminho até o arquivo criptografado (será necessária a senha da chave privada):

```
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 ...
```

Também é possível especificar a opção `--output` para salvar a mensagem em um novo arquivo não criptografado:

```
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 ...
```

Assinatura e verificação de arquivos

Além de criptografar, o GPG também pode ser usado para assinar arquivos. A opção `--sign` serve para isso. Vamos começar criando uma nova mensagem (`message`) e assinando-a com a opção `--sign` (a senha de sua chave privada será solicitada):

```
carol@debian:~$ echo "This is the message to sign ..." > message
carol@debian:~$ gpg --output message.sig --sign message
(...)
```

Vamos explicar melhor o comando `gpg`:

`--output message`

Especificação do nome de arquivo da versão assinada do arquivo original (`message.sig` em nosso exemplo).

`--sign message`

Caminho para o arquivo original.

NOTE Com `--sign`, o documento é compactado e depois assinado. A saída será em formato binário.

Em seguida, transferimos o arquivo para `ina` em `halof` usando `scp message.sig ina@halof:/home/ina`. De volta a `ina` em `halof`, podemos verificar o resultado com a opção `--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]
(...)
```

Se quiser ler o arquivo, você precisa descriptografá-lo em um novo arquivo (`message`, em nosso caso) usando a opção `--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 ...
```

GPG-Agent

Vamos encerrar esta lição com algumas palavras sobre o `gpg-agent`. `gpg-agent` é o daemon que gerencia as chaves privadas para o GPG (ele é iniciado sob demanda pelo `gpg`). Para ver um resumo das opções mais úteis, execute `gpg-agent --help` ou `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:

<code>--daemon</code>	run in daemon mode (background)
<code>--server</code>	run in server mode (foreground)
<code>--supervised</code>	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 Para saber mais, consulte a página de manual do gpg-agent.

Exercícios Guiados

1. Complete a tabela com o nome de arquivo correto:

Descrição	Nome de arquivo
Banco de dados de confiabilidade	
Diretório dos certificados de revogação	
Diretório das chaves privadas	
Chaveiro público	

2. Responda às questões a seguir:

- Que tipo de criptografia é usado pelo *GnuPG*?

- Quais os dois componentes principais da criptografia de chave pública?

- Qual o KEY-ID da impressão digital de chave pública 07A6 5898 2D3A F3DD 43E3 DA95 1F3F 3147 FA7F 54C7?

- Qual o método usado para distribuir chaves públicas em nível global?

3. Coloque as seguintes etapas na ordem correta para a revogação de uma chave privada:

- Tornar a chave revogada disponível para seus correspondentes.
- Criar um certificado de revogação.
- Importar o certificado de revogação para seu chaveiro.

A ordem correta é:

Passo 1:	
Passo 2:	
Passo 3:	

4. Quando falamos em criptografia de arquivos, o que a opção `--armor` significa no comando `gpg --output encrypted-message --recipient carol --armor --encrypt unencrypted-`

message?

Exercícios Exploratórios

1. A maioria das opções do gpg têm uma versão longa e uma curta. Complete a tabela com a versão curta correspondente:

Versão longa	Versão curta
--armor	
--output	
--recipient	
--decrypt	
--encrypt	
--sign	

2. Responda às seguintes questões sobre a exportação de chaves:

- Qual comando você usaria para exportar todas as suas chaves públicas para um arquivo chamado all.key?

- Qual comando você usaria para exportar todas as suas chaves privadas para um arquivo chamado all_private.key?

3. Qual opção do gpg permite realizar a maioria das tarefas de gerenciamento de chaves por meio de um menu?

4. Qual opção do gpg permite fazer uma assinatura em texto não criptografado?

Resumo

Esta lição tratou do *GNU Privacy Guard*, uma excelente opção para criptografar/descriptografar e assinar/verificar arquivos digitalmente. Você aprendeu:

- como gerar um par de chaves.
- como listar as chaves em seu chaveiro.
- o conteúdo do diretório `~/.gnupg`.
- o que são `USER-ID` e `KEY-ID`.
- como distribuir chaves públicas para seus correspondentes.
- como distribuir chaves públicas globalmente por meio de servidores de chaves.
- como revogar chaves privadas.
- como criptografar e descriptografar arquivos.
- como assinar e verificar arquivos.
- o básico sobre o *GPG-Agent*.

Os seguintes comandos foram discutidos nesta lição:

gpg

ferramenta de criptografia e assinatura do *OpenPGP*.

Respostas aos Exercícios Guiados

1. Complete a tabela com o nome de arquivo correto:

Descrição	Nome de arquivo
Banco de dados de confiabilidade	trustdb.gpg
Diretório dos certificados de revogação	opengp-revocs.d
Diretório das chaves privadas	private-keys-v1.d
Chaveiro público	pubring.kbx

2. Responda às questões a seguir:

- Que tipo de criptografia é usado pelo *GnuPG*?

Criptografia de chave pública ou criptografia assimétrica.

- Quais os dois componentes principais da criptografia de chave pública?

A chave pública e a chave privada.

- Qual o KEY-ID da impressão digital de chave pública 07A6 5898 2D3A F3DD 43E3 DA95
1F3F 3147 FA7F 54C7?

FA7F 54C7

- Qual o método usado para distribuir chaves públicas em nível global?

Servidores de chave.

3. Coloque as seguintes etapas na ordem correta para a revogação de uma chave privada:

- Tornar a chave revogada disponível para seus correspondentes.
- Criar um certificado de revogação.
- Importar o certificado de revogação para seu chaveiro.

A ordem correta é:

Passo 1:	Criar um certificado de revogação.
Passo 2:	Importar o certificado de revogação para seu chaveiro.

Passo 3:

Tornar a chave revogada disponível para seus correspondentes.

4. Quando falamos em criptografia de arquivos, o que a opção `--armor` significa no comando `gpg --output encrypted-message --recipient carol --armor --encrypt unencrypted-message?`

Ela produz uma saída ASCII blindada, que permite copiar o arquivo criptografado resultante em um email.

Respostas aos Exercícios Exploratórios

1. A maioria das opções do gpg têm uma versão longa e uma curta. Complete a tabela com a versão curta correspondente:

Versão longa	Versão curta
--armor	-a
--output	-o
--recipient	-r
--decrypt	-d
--encrypt	-e
--sign	-s

2. Responda às seguintes questões sobre a exportação de chaves:

- Qual comando você usaria para exportar todas as suas chaves públicas para um arquivo chamado all.key?

gpg --export --output all.key ou gpg --export -o all.key

- Qual comando você usaria para exportar todas as suas chaves privadas para um arquivo chamado all_private.key?

gpg --export-secret-keys --output all_private.key ou gpg --export-secret-keys -o all_private.key (--export-secret-keys pode ser trocado por --export-secret-subkeys com um resultado ligeiramente diferente — confira man pgp para saber mais).

3. Qual opção do gpg permite realizar a maioria das tarefas de gerenciamento de chaves por meio de um menu?

--edit-key

4. Qual opção do gpg permite fazer uma assinatura em texto não criptografado?

--clearsign

Imprint

© 2023 by Linux Professional Institute: Materiais Didáticos, “LPIC-1 (102) (Version 5.0)”.

PDF gerado: 2023-07-13

Este trabalho está licenciado sob Creative Commons Licença Atribuição-NãoComercial-SemDerivações 4.0 Internacional (CC BY-NC-ND 4.0). Para ver uma cópia desta licença, visite

<https://creativecommons.org/licenses/by-nc-nd/4.0/>

Embora o Linux Professional Institute tenha agido de boa fé para garantir que as informações e instruções contidas neste trabalho sejam exatas, o Linux Professional Institute isenta-se de qualquer responsabilidade por erros ou omissões, incluindo, sem limitações, a responsabilidade por danos resultantes do uso ou confiança nesta obra. O uso das informações e instruções contidas neste trabalho deve ser feito por sua própria conta e risco. Se as amostras de código ou outras tecnologias contidas ou descritas neste trabalho estiverem sujeitas a licenças de código aberto ou direitos de propriedade intelectual de terceiros, é sua responsabilidade garantir que seu uso esteja em conformidade com tais licenças e/ou direitos.

Os Materiais Didáticos da LPI são uma iniciativa do Linux Professional Institute (<https://lpi.org>). Os Materiais Didáticos e suas traduções estão disponíveis em <https://learning.lpi.org>.

Para perguntas e comentários sobre esta edição, bem como sobre todo o projeto, escreva para: learning@lpi.org.