Controle de usuários

Muitas tarefas são mais rápidas e simples com uma interface de texto, e administrar usuários não é uma exceção. Ajudamos você a se familiarizar com os arquivos e comandos necessários para adicionar, modificar e remover contas de usuários e de grupos.

Matt Simmons

Na Idade Média, quando máquinas eram medidas em metros quadrados, o conceito de múltiplas pessoas usando um computador ao mesmo tempo era quase inconcebível. Conforme a eletrônica progrediu, o conceito de "tempo compartilhado" reinou. Programas (*jobs*) eram submetidos e executados, e os resultados coletados eram enviados de volta para o programador. Em seguida, o job da próxima pessoa era submetido. Uma corrida morro acima nos dois lados.

Nos anos 1970, o compartilhamento de tempo cedeu a vez ao *multitasking* (multitarefa), com múltiplos programas sendo executados ao mesmo tempo. Isso levou ao conceito de contas individuais de usuários, nas quais várias pessoas podiam fazer *login* na máquina e acessar seus próprios ambientes. Este avanço necessitava de segurança de usuários, assim como ferramentas para gerenciar essas contas. Embora a Ciência da Computação tenha avançado dramaticamente desde essa época, o conceito da computação multiusuário veio para ficar.

Fundamentos de contas

O Linux é um sistema operacional multiusuário que suporta tanto usuários quanto grupos. Cada conta de usuário existe no sistema como duas linhas em um par de arquivos: uma em `/etc/passwd` e uma correspondente em `/etc/shadow`. Cada usuário recebe um identificador único, chamado UID (*User ID*). De forma semelhante, cada grupo consiste em uma linha em `/etc/group` e recebe um identificador de grupo, chamado de GID (*Group ID*). UIDs e GIDs válidos são inteiros de 0 a 65536, apesar de o máximo recomendado ser 65535. UIDs entre 0 e 999 geralmente são reservados para contas do sistema. A conta do superusuário, conhecido como *root*, deve sempre ter UID 0 e pertencer a um grupo também conhecido como *root*, que recebe o GID 0. Isso garante que certas suposições feitas pelo sistema estejam corretas.

O arquivo `/etc/passwd` pode ser lido por todos os usuários da máquina. Como permitir que os usuários vejam senhas – mesmo que estejam criptografadas – é um risco à segurança, elas foram retiradas do arquivo `passwd` e transferidas para o `/etc/shadow`, que somente o superusuário pode ler.

Cada linha é separada em campos por dois pontos (**figura 1**), com o primeiro campo consistindo no nome de usuário que identifica o dono da conta na máquina. Um símbolo é guardado no segundo campo, geralmente um `x`, caso a conta tenha permissão de login, ou um * se o login for proibido para ela. Esse campo originalmente continha a senha do usuário, mas boas práticas de segurança o retiraram do arquivo

Figura 1: Exemplo de entrada no arquivo `/etc/passwd`.

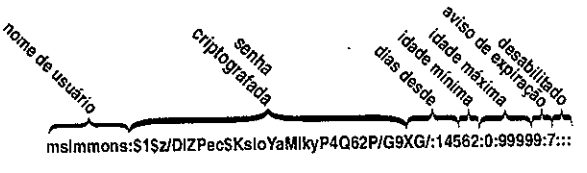

Figura 2: Exemplo de entrada no arquivo `/etc/shadow`.

`/etc/passwd`, legível por todos, e o colocaram no `/etc/shadow`.

O terceiro e o quarto campos são o UID e o GID, respectivamente. O quinto campo é chamado de GECOS, que significa *General Electric Comprehensive Operating System*. Historicamente, esse campo era usado para armazenar números de tags de segurança; agora, é usado para armazenar outras informações sobre um usuário em um formato separado por vírgulas. Às vezes, ele é chamado de "campo de comentário", e apesar de em muitos casos guardar apenas o nome do usuário, o formato completo é: Nome do usuário, número do prédio/sala, telefone comercial, outro telefone.

O sexto campo do arquivo é o diretório `home` da conta. Quando um usuário faz login, as preferências do usuário serão guardadas no diretório informado nesse campo. Se o diretório informado não existir, o usuário fará login com / como seu diretório base.

O último campo do arquivo `/etc/passwd` é o Shell do usuário. Quando os usuários fazem login, eles precisam de um interpretador de comandos para interagir com o sistema e executar comandos. Este campo informa qual é esse ambiente. Se você examinar seu próprio arquivo `/etc/passwd`, notará que várias contas usam `/sbin/nologin` como Shell. Isso ajuda a evitar que essas contas recebam acesso a um ambiente de Shell caso os processos nos quais são executadas estejam comprometidos.

Assim como o arquivo `/etc/passwd`, o `/etc/shadow` também é delimitado por dois pontos, com uma linha por conta (**figura 2**). Como antes, o primeiro campo é o nome do usuário da conta, e o segundo é a senha criptografada. O método de criptografia depende da distribuição Linux. Originalmente era usado um esquema de criptografia chamado `crypt`, mas com o advento de computadores mais poderosos, um método de `hash` `md5` foi escolhido por ser mais difícil de quebrar.

Os campos do terceiro ao sexto no arquivo `/etc/shadow` são usados para informações de idade da senha. As melhores práticas de políticas de segurança requerem alterações regulares de senhas por meio de uma idade máxima da senha. Para evitar que os usuários reutilizem senhas para retornar a sua senha original, diversas políticas de segurança determinam também uma idade mínima para elas. Esses campos facilitam estas políticas. Um ponto importante é que cada uma dessas exigências se aplica somente ao usuário em questão; o superusuário pode alterar as senhas dos usuários sem qualquer restrição.

O terceiro campo do `/etc/shadow` indica a idade da senha e contém o número de dias desde o início do tempo do Unix (1 de janeiro de 1970). Para traduzir esse número para uma data, use o comando `date` (**listagem 1**). No exemplo, isto indica que a senha do usuário *msimmons* foi definida em 8 de novembro de 2009. O restante dos campos de datas no arquivo é relativo a este número.

O quarto campo do arquivo denota o número mínimo de dias antes dos quais a senha pode ser alterada. No caso da minha conta de usuário, o número é 0, indicando que não há um número mínimo de dias, e a senha pode ser alterada imediatamente.

O quinto campo especifica a idade máxima da senha. Nenhuma opção especifica que este idade seja ilimitada, mas cinco 9s indicam uma faixa de tempo de quase 280 anos. Se você alterar sua senha amanhã, a data da próxima alteração obrigatória pode ser uma data estelar.

O sexto campo armazena quantos dias antes da expiração da senha o usuário deve ser avisado. Em nosso exemplo, sete dias antes da senha expirar será enviado um aviso, ao fazer login, informando que a senha expirará em uma semana, e o usuário terá a opção de alterá-la nesse momento.

O sétimo campo especifica quantos dias após a expiração da senha a conta será desativada. Isto impede que senhas antigas representem riscos de segurança. Uma vez desativada a conta, o oitavo campo armazenará o número de dias desde o início do tempo do Unix que a conta foi desativada. O último campo atualmente está reservado para uso futuro.

O conteúdo do arquivo `/etc/group` é estruturado de forma semelhante aos arquivos `passwd` e `shadow`, embora mais simples na sua composição. O primeiro campo é o nome do grupo, seguido de um símbolo no estilo do arquivo `passwd`. O terceiro campo é o GID, e o último campo é uma lista separada por vírgulas de contas de usuários pertencentes a este grupo.

Listagem 1: Tradução de uma data do Unix

```
01 $ grep msimmons /etc/shadow
02 msimmons:$1$UVHO7Lj9$SguVZINtWWoCo17
   ↪m2uSe/1:14556:0:99999:7:::
03 $ date -d "Jan 1 1970 + 14556 days"
04 Dom Nov  8 00:00:00 BRST 2009
```

Quando um GID é especificado para um usuário no arquivo `/etc/passwd`, não é necessário definir explicitamente, no arquivo `/etc/group`, que o usuário pertence a esse grupo.

Para criar um novo grupo, é usado o comando `/usr/sbin/groupadd <nome do grupo>` e, se o comando tiver sucesso, nenhuma saída é gerada. Também há opções disponíveis para especificar o GID do novo grupo com `groupadd -g <GID>`.

Existem várias ferramentas para examinar os usuários do sistema. A mais simples é o comando `finger`. Ao ser chamado com um nome de usuário como argumento, ele imprime várias informações de uma vez. Os vários campos incluem informações sobre o usuário obtidas nos arquivos já citados, assim como vários *logs* e arquivos no diretório `home` do usuário. Historicamente, o comando `finger` pode examinar um usuário local ou conversar com um servidor `finger` executado na porta 79 do protocolo TCP de uma máquina Unix remota. As boas práticas de segurança gradativamente retiraram esse serviço de uso a ponto de não fazer parte de nenhuma instalação padrão conhecida, embora ainda seja possível configurá-lo.

Para consultar os usuários atualmente usando o sistema, use o comando `who` ou, caso prefira uma saída mais detalhada, o comando `w` (**listagem 2**). Ambos exibem o nome do usuário e o dispositivo ao qual o usuário está conectado, mas o `w` mostra várias métricas de desempenho além dos programas que os usuários estão executando.

Listagem 2: Utilizar who ou w?

```
01 $ who
02 msimmons tty1 2009-11-08 14:58
03 root :0 2009-11-08 12:46
04 root pts/0 2009-11-08 12:48 (:0.0)
05 msimmons pts/2 2009-11-08 14:58 (centos)
06
07 $ w
08 15:00:51 up 8:58, 4 users, load average: 0.07, 0.10,0.10
09 USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
10 msimmons tty1 - 14:58 2:19 0.22s 0.22s -bash
11 root :0 - 12:46 ?xdm? 4:11 0.92s /usr/bin/gnome-session
12 root pts/0 :0.0 12:48 0.00s 0.76s 0.29s ssh msimmons@localhost
13 msimmons pts/2 centos 14:58 0.00s 0.20s 0.02s w
```

Adicionar usuários e grupos

Estritamente falando, é totalmente possível adicionar, modificar e remover usuários por meio de editores de texto. Porém, existem várias ferramentas para fazer isso, o que prova que essa não é uma atividade divertida. Os utilitários mais simples para criar usuários e grupos são o comando `useradd` e o `groupadd`, respectivamente. Eles residem no diretório `/usr/sbin` por padrão e, para usá-los, é preciso estar logado como superusuário.

O uso do `useradd` é simples e direto, embora ele possua várias opções. Ao usá-lo pela primeira vez, a opção mais útil é `useradd -D` (**listagem 3**). A opção `-D` exibe os valores padrão de várias opções, conforme determinadas pelo arquivo `/etc/default/useradd`.

Diferentes distribuições possuem configurações distintas, então, verifique o conteúdo do arquivo antes de fazer suposições incorretas. Uma forma ainda mais segura seria informar, explicitamente, as configurações de usuários na linha de comando. Assim como em várias coisas no Linux, a melhor técnica é aquela que respeita o seu nível de conforto.

Criar um novo usuário na linha de comando é simples. Digitar o comando `useradd <nome do usuário>` é suficiente para adicionar as linhas nos arquivos `/etc/passwd` e `/etc/shadow`. Contudo, a conta não é incrivelmente útil neste estado, pois não foi atribuída nenhuma senha e o diretório `home` do usuário não foi criado. Para criar uma conta mais útil, lance mão de vários argumentos de linha de comando que ajustam a conta de usuário às suas necessidades. Por exemplo, eu poderia criar minha conta da seguinte forma:

```
# /usr/sbin/useradd msimmons \
    -c "Matt Simmons" \
    -d /home/msimmons \
    -m -s /bin/bash
```

Além de criar as linhas no `passwd` e no `shadow`, isto inclui Matt Simmons no campo GECOS (`-c`), especifica meu diretório `home` (`-d`), cria o diretório especificado (`-m`) e define meu Shell como `/bin/bash` (`-s`). Desta forma, a conta se torna muito mais útil.

Uma opção do `useradd` que ainda não usamos é a `--password`, que aceita um hash de senha criptografado como argumento. Pessoalmente, eu tento evitar esta opção porque ela oferece uma curta janela de tempo na qual a senha criptografada fica disponível para todos os usuários do sistema, e porque é preciso se dar ao trabalho de obter a senha criptografada. Se a criação dos usuários for feita por meio de um script e os novos usuários tiverem uma mesma senha padrão, esse recurso pode ser útil, embora talvez seja melhor dar uma olhada no comando `/usr/sbin/newusers`, que permite a importação de contas de usuários em lote.

Listagem 3: Inclusão de usuários

```
01 # useradd -D
02 GROUP=100
03 HOME=/home
04 INACTIVE=-1
05 EXPIRE=
06 SHELL=/bin/bash
07 SKEL=/etc/skel
08 CREATE_MAIL_SPOOL=yes
```

A única desvantagem do comando newusers é que as senhas são armazenadas em texto puro e oferecem mais um vetor de ataque caso o arquivo seja desprotegido.

Se você não usa a opção --password, a conta é desativada e sua senha precisará ser gerenciada com o comando passwd. Para criar uma senha para um novo usuário, use /usr/bin/passwd <nome do usuário>. Muitas distribuições possuem uma complexidade de senha padrão que impede que um usuário as viole, mas permite isso ao *root* com um simples aviso.

Modificar usuários e grupos

Após adicionar usuários ao sistema, mantê-los pode se tornar uma tarefa em tempo integral. Para ajudar nisso, há várias ferramentas disponíveis para examinar e alterar contas e grupos. A ferramenta primária para modificar contas de usuários já existentes é /usr/sbin/usermod, que aceita todas as opções de linha de comando do useradd, inclusive a alteração do nome do usuário (--login <NOVO-NOME>) e UID (--uid <NOVO-UID>).

O comando /usr/sbin/groupmod existe para modificar configurações individuais de grupos. Assim como o usermod, ele suporta todas as opções do comando groupadd exibidas anteriormente.

O comando usermod pode modificar o campo GECOS/comentário mas deixa a formatação por conta de quem o utilizar. O comando /usr/bin/chfn dá acesso muito mais fácil ao formato padronizado e, sem opções, até oferece prompts para preencher todos os campos.

O comando passwd também pode alterar algumas (mas não todas) configurações de idade da senha, mas a ferramenta /usr/bin/chage tem total controle sobre cada opção. Como o mecanismo de idade das senhas é complexo, o chage aceita datas assim como números de dias desde o início do tempo Unix. A **listagem 4** mostra detalhes da alteração da expiração de senha que usa uma data formatada de forma comum em vez do número de dias desde o início do tempo Unix.

Ambiente do usuário

Na **listagem 3**, você pôde notar que o SKEL na linha 7, possui o valor padrão /etc/skel, que indica o diretório skeleton (esqueleto) a partir do qual os diretórios home dos novos usuários devem ser criados.

Sempre que uma nova conta é criada, esse diretório age como modelo, e todos os arquivos e estruturas de diretórios que existem nele são criados no diretório home do usuário especificado no comando useradd. Este método é ideal para distribuir um perfil padrão ou um mesmo ambiente Bash com cada novo usuário.

Outro arquivo necessário para administrar ambientes de usuários é o /etc/profile. Cada vez que um usuário faz login, o /etc/profile é carregado, e qualquer alteração a ser realizada no ambiente é efetuada imediatamente. Geralmente, essas alterações definem vários padrões de sistema, tais como textos de prompt, caminhos de execução, umasks e outras.

Tão importante quanto conseguir modificar contas de usuários é conseguir modificar o ambiente dos usuários. Em um computador utilizado por múltiplos usuários, garantir que os recursos do sistema estejam disponíveis para todos é importante, e uma das armas mais potentes nesse sentido é o parâmetro ulimit embutido no Bash.

Cada ação tomada em um sistema Linux usa certa quantidade de recursos do sistema. Esses recursos são finitos, e quando muitos usuários estão em contenção, garantir a distribuição justa é vital para um sistema bem administrado. Com isso em vista, o ulimit controla várias limitações de disco, memória e processos, para garantir que nenhum usuário seja capaz de sobrecarregar o sistema. Este comando é executado a partir de um Shell de usuário e afeta tanto o Shell quanto todos os programas gerados por ele.

O comando ulimit aceita limites hard e soft, além de valores sem limite. Uma vez definido, um limite hard é imutável. Um limite soft inicial pode ser redefinido com valor máximo igual ao limite hard, mas jamais superior a ele. Isto oferece um método de aumentar os recursos dinamicamente, sempre que necessário. Uma configuração "ilimitada" não oferece qualquer proteção e permite que o Shell do usuário consuma todos os recursos daquele tipo particular.

Como é relativamente fácil proibir atividades de usuário permissivas demais, é importante entender as implicações de ajustar os limites com o ulimit. Por exemplo, ao limitar o tamanho de memória, é possível impedir que os usuários carreguem programas legítimos que utilizam bibliotecas compartilhadas.

A aplicação típica do ulimit na administração de usuários, é definir limites hard e soft no arquivo /etc/profile. Como esse arquivo é carregado toda vez que um usuário faz login, qualquer limite hard definido nele será imutável pelo usuário. Os

Listagem 4: Uso do comando chage

```
01 # cat /etc/shadow | grep msimmons ➥
02 msimmons:$1$MfBYOD/O$yv2.31xLwrsN9oXt1Bxpa0:14563:0:99999:7:::
03 # chage -E "2009-11-16" msimmons
04 # cat /etc/shadow | grep msimmons ➥
05 msimmons:$1$MfBYOD/O$yv2.31xLwrsN9oXt1Bxpa0:14563:0:99999:7::14564:
06 # date -d "Jan 1, 1970 + 14564 days"
07 Dom Nov  8 00:00:00 BRST 2009
```

limites `soft` podem ser ajustados conforme necessário, até os limites definidos para eles.

Embora esteja fora do escopo deste artigo cobrir cada propriedade definível pelo `ulimit`, há várias configurações particularmente interessantes para administradores de sistemas com limites restringidos. Se você tiver espaço em disco limitado, o tamanho "core" de arquivos (`-c`) e o tamanho real de arquivos (`-f`) podem evitar que seus usuários criem acidentalmente (ou intencionalmente) arquivos grandes demais que consumam todo o espaço disponível. Se você ou seus usuários forem testar softwares experimentais, definir um número máximo de processos (`-u`), tempo de CPU (`-t`) e tamanho máximo de memória (`-m`) podem ser boas proteções contra processos fugitivos que tenderiam a sobrecarregar a máquina.

Para verificar os limites atuais, use um comando `ulimit -a` (**listagem 5**). Cada uma das configurações é exibida com seu nome em texto puro na coluna da esquerda, a opção que a altera no meio e o limite atual na direita. Para definir tanto limites `hard` quanto `soft`, use `ulimit <opção> <limite>`. Para definir apenas um limite `hard`, use `-H` e, para apenas um limite `soft`, `-S`.

Embora normalmente seja usado apenas para alterar ou definir senhas de usuários, o comando `/sbin/passwd` possui várias opções avançadas que podem ser usadas para gerenciar contas. Por exemplo, a opção `-e` expira senhas imediatamente, `-l` bloqueia (*lock*) uma conta, `-u` desbloqueia (*unlock*) e `-S` exibe o status da conta.

Excluir usuários e grupos

Contas de usuários, assim como todos os aspectos dos computadores, possuem ciclos de vida. Em algum momento, as contas precisarão ser excluídas da máquina, e é melhor para todos os envolvidos se os vestígios das contas há muito esquecidas forem, de fato, apagados.

Em um ambiente corporativo, é importante revisar as políticas de conservação de dados da empresa e garantir que todos os backups necessários sejam feitos antes de eliminar os dados dos usuários da máquina. Se os dados precisarem ser conservados, é preciso decidir quem fica com os arquivos e onde eles serão armazenados – isto é, se uma simples mudança na propriedade do arquivo é suficiente ou é necessário extraí-lo para uma mídia *offline* e apresentá-lo em formato físico.

Após essas questões serem resolvidas, o primeiro passo na remoção total de uma conta de usuário é tomar nota do seu UID e também do grupo ao qual ele pertence. A forma mais simples de fazer isso é com o comando `id`.

```
# id msimmons
uid=1002(msimmons)
➥ gid=1004(msimmons)
➥ groups=1004(msimmons),
➥37(operator)
```

De posse destas informações, será possível encontrar os arquivos de propriedade do usuário *msimmons* em todos os sistemas de arquivos e atribuí-los aos usuários adequados. Neste caso, meu UID é 1002, então o seguinte comando `find` pode localizar todos os arquivos de propriedade da minha conta: `# find / -uid 1002`.

Com o `find`, é possível realizar inúmeras tarefas, tais como apagar, fazer `chmod` (modificar as permissões de acesso) ou copiar arquivos, conforme necessário. Supondo que os arquivos do usuário sejam incluídos, a conta é apagada com `/usr/sbin/userdel <nome do usuário>`. Com a opção `-r` é possível apagar automaticamente o diretório `home` do usuário, mas os arquivos de sua propriedade que não se encontrarem no diretório `home` precisarão ser tratados manualmente pelo `find`.

Apagar o grupo do usuário é simples: `/usr/sbin/groupdel <nome do grupo>`. Isto apaga a linha do `/etc/group` mas não altera nenhum arquivo de propriedade desse grupo. Uma forma melhor de lidar com esses arquivos é usar o `find`, substituindo o `-uid` usado anteriormente pelo `-gid`.

Conclusão

A administração de usuários é uma parte importante da convivência com sistemas Linux. A familiaridade com ferramentas de linha de comando aumenta sua eficiência e sua flexibilidade – para não falar da confiança – quando não há uma interface gráfica disponível. As ferramentas cobertas neste artigo oferecem uma base sólida para seu conjunto de administração. ■

Listagem 5: Comando ulimit -a

```
01 $ ulimit -a
02 core file size (blocks, -c) 0
03 data seg size (kbytes, -d) unlimited
04 scheduling priority (-e) 0
05 file size (blocks, -f) unlimited
06 pending signals (-i) 32768
07 max locked memory (kbytes, -l) unlimited
08 max memory size (kbytes, -m) unlimited
09 open files (-n) 1024
10 pipe size (512 bytes, -p) 8
11 POSIX message queues (bytes, -q) 819200
12 real-time priority (-r) 0
13 stack size (kbytes, -s) 8192
14 cpu time (seconds, -t) unlimited
15 max user processes (-u) 32768
16 virtual memory (kbytes, -v) unlimited
17 file locks (-x) unlimited
```