Controlo de acesso ao sistema de ficheiros em Linux

João Nuno Gomes Rodrigues de Almeida
A75209 - Universidade do Minho

João Miguel Matela Aidos Manso de Araújo
A75364 - Universidade do Minho

Controlo de acesso ao sistema de ficheiros em Linux: mecanismo tradicional e extensões, utilizadores e grupos de utilizadores reais e efetivos, elevação de privilégios.

Introdução

Com este ensaio pretendemos explorar um pouco o mundo da segurança em Unix, nomeadamente o sistema de gestão de múltiplos utilizadores, controlo de acesso e outros. Optámos por escolher este tema, pois achámos de extrema importância abordar a questão das vulnerabilidades e situações incómodas que podem surgir, caso os incorretos privilégios de controlo de ficheiros sejam concedidos a um dado grupo de utilizadores. Até mesmo situações onde vulnerabilidades pouco visíveis, podem ser exploradas pelos prevaricadores após serem concedidos dados privilégios no controlo do sistema de ficheiros, que aparentemente não levantariam quaisquer problemas de segurança.

Inicialmente iremos explorar um pouco os conceitos básicos de controlo de acesso necessários para a compreensão básica do documento [3]. Iremos depois passar a mecanismos explícitos que um sistema Linux implementa para assegurar a implementação segura de um sistema de controlo de acesso, quer ao nível dos utilizadores do sistema e os seus conjuntos/grupos, quer ao nível dos próprios ficheiros envolvidos. Por último, iremos brevemente referir alguns problemas de segurança que atualmente atingem a arquitetura, como se encontra implementada.

Conceitos básicos

Definições Elementares

Para uma melhor perceção do conceito de direitos de acesso ao sistema de ficheiros é importante também definir alguns termos extremamente importantes e regularmente usados.

Subject - Entidade que pretende efetuar ações de acesso sobre um objeto;

Object - Entidade (ficheiros ou recursos) com um ou mais atributos de controlo de acesso a si atribuídos, que ditam o acesso de um ou mais *subjects* no sistema;

Group - Conjunto de *subjects*. Esta definição é especialmente útil na atribuição dos mesmos direitos de acesso a mais do que um *subject*.

Modos de acesso em Linux

A implementação dos sistemas de controlo de acesso em Linux baseia-se, em parte, no modelo de Bell-LaPadula, com três modos de acesso distintos, sendo que estes aplicam-se tanto a ficheiros como a pastas:

- Read
  - Ficheiros - ver conteúdos do ficheiro;
  - Pastas - listar conteúdos da diretoria.
- Write
  - Ficheiros - escrever para um ficheiro;
  - Pastas - criar ou mudar o nome de um ficheiro na diretoria.
- Execute
  - Ficheiros - executar um ficheiro;
  - Pastas - percorrer a diretoria.

Observe-se que esta notação simples em Linux contém vários detalhes, não óbvios à primeira partida. Como só temos um tipo para modificar ficheiros, o *write*, operações de criação e eliminação das mesmas ficam delegadas a este direito de acesso. Por outro lado, se um dado ficheiro tiver a si atribuído uma permissão de escrita, a permissão de leitura não é implicitamente dada, pois Linux permite operações de acesso de escrita exclusivas.

ACL

Qualquer Sistema Operativo (SO) que implemente mecanismos de controlo de acesso necessita de ter também definida uma Access Control List (ACL). Esta gere os direitos de acesso dos vários objetos, sendo que entradas numa ACL são denominadas Access Control Entries (ACE). Cada uma destas entradas tem a seguinte estrutura genérica:

```
Object 1: User 1 (read); User 2 (read, write, execute)
Object 2: Group 1 (read, write)
Object 3: User 2 (read, execute)
...
```

Sistema de ficheiros Linux

De seguida irão ser abordados, três aspetos importantes no contexto de funcionamento do sistema de ficheiros do Linux: os utilizadores e seus grupos, as permissões concedidas a esses utilizadores, de que forma o Linux, através das permissões que concede, garante a segurança dos ficheiros, terminando finalmente com as vulnerabilidades em todo esse processo.

O sistema de ficheiros do Linux é constituído, essencialmente, por dezasseis diretorias [5], localizadas na raiz do sistema, ou root (/). Cada uma destas diretorias tem um propósito no sistema, e desempenha algum tipo de papel no funcionamento do SO.

- / - Raiz do sistema;
- /bin - Programas usados no arranque, ou reparação do sistema;
- /boot - Inicialização do kernel do Linux;
- /dev - Dispositivos como hardisk, floopy, etc;
- /etc - Ficheiros de configuração e inicialização;
- /sbin - Administração e reparação do SO;
- /home - Diretorias de trabalho dos utilizadores;
- /lib - Bibliotecas para o correto funcionamento do sistema e programas do mesmo;
- /mnt - Montagem de unidades amovíveis (USB, CDs,...);
- /opt - Programas não oficiais;
- /proc - Ficheiros virtuais, do estado atual dos processos e componentes do SO;
- /tmp - Ficheiros temporários;
- /usr
  - /usr/local - Instalação de outros programas;
  - /usr/bin - Comandos destinados aos utilizadores;
  - /usr/src - Código fonte do SO, modificável;
  - /usr/lib - Bibliotecas relacionadas com a programação;
- /var - Serviços variados;
  - /var/spool - Filas de espera das impressoras e sistemas de correio eletrónico;
  - /var/lock - Utilização atual de serviços;
  - /var/log - Registo de todos os eventos que ocorrem no sistema (particularmente útil na área de segurança).

Utilizadores

Dentro do Linux, é aceitável uma divisão em quatro partes no que toca à categorização dos utilizadores [2, 7]:

- **Super-Utilizador** - Este utilizador é designado como *root*. É responsável por todo o sistema e não tem qualquer tipo de restrição acerca do que pode ou não fazer;
- **Utilizador do sistema** - Este tipo de utilizadores não precisam de estar autenticados ou identificados pelo sistema para executarem determinadas tarefas, como por exemplo, o controlo de servidores *Apache*;
- **Utilizador comum** - Este tipo de utilizadores foram criados pelo super-utilizador e possuem contas para aceder às capacidades do SO. Estes utilizadores podem executar tarefas de criação e manipulação, consoante as suas permissões. Trabalham num ambiente próprio (/home/username/) e podem ser identificados de três maneiras:
  - Login - Designação escolhida para a identificação do utilizador, quando mapeada com a *password*;
  - UID - Número atribuído, no momento da criação do utilizador, para que o mesmo possa ser definido inequivocamente, pelas aplicações/programas;
  - Nome - Designação que o utilizador indicou no momento de criação;
- **Grupos** - Conjunto de utilizadores que partilham um conjunto de permissões.

Permissões

A partir do momento em que um sistema é partilhado por mais do que um utilizador, passa a ser necessário que se estabeleça um método para controlar a manipulação de ficheiros, tendo em conta o seu dono ou a sua importância no funcionamento do sistema operativo.

Tradicionalmente, o modo como os sistemas operativos Linux procedem ao controlo de acesso aos ficheiros baseia-se em conceder permissão para a execução de um conjunto de três ações (*read*, *write*, *execute*) a um determinado utilizador, ou grupo, para cada um dos ficheiros do sistema.

O conjunto de permissões dado a um utilizador e/ou grupo tem um propósito, e existe uma intenção para que aquele conjunto tenha sido dado e não um outro qualquer.

Essencialmente este tipo de permissões é dado, ou retirado, consoante o tipo de ficheiros que se trata. Deve ser estipulado se existe ou não a necessidade urgente de proteger os mesmos de algum erro que o utilizador possa acidentalmente cometer que comprometa o correto funcionamento do sistema, ou algum ato propositado que o mesmo possa cometer com o único intuito de prejudicar ficheiros que não são seus, ou o próprio sistema, explorando as suas vulnerabilidades.

Por exemplo, no caso dos ficheiros do SO, é altamente aconselhável que não se alterem nenhumas das permissões estabelecidas sobre o risco da criação de instabilidade na máquina.

O super-utilizador (*root*) é o único utilizador que tem todas as permissões disponíveis para qualquer tipo de ficheiro, podendo fazer o que bem entende com qualquer um.

O mecanismo clássico de como esta gestão de permissões é feita consiste na existência de um controlo de acesso para cada ficheiro no sistema, controlo este dividido em três classes ou secções [6]. A consulta dos controlos de acesso de um determinado utilizador para um conjunto de ficheiros pode ser feita através do comando "ls -l".

Tipo	Dono	Grupo	Outro
d	r w x	r w x	r w x

O tipo designa o tipo de ficheiro ao qual o controlo de acesso que se segue (num formato binário de 12 bits ou 9 caracteres) está associado.

- d - Diretoria;
- - - Ficheiro;
- l - Link;
- s - Socket;
- p - Pipe;
- b - Bloco;
- c - Caracteres.

As três classes que se seguem: dono, grupo e outro, definem as permissões que o dono do ficheiro tem sobre o mesmo, as permissões do grupo de utilizadores a que ele pertence e, por último, as permissões que outros utilizadores têm, não sendo donos nem pertencendo ao mesmos grupo que o dono do ficheiro.

No que diz respeito às permissões em si, o seu significado varia caso estejamos a tratar de ficheiros ou diretorias, tal como já foi abordado na secção introdutória.

A representação binária de cada um dos controlos de acesso, referentes a um ficheiro ou diretoria, é feita com recurso a 12 bits [1].

	Dono	Grupo	Outro
0 0 0	1 1 1	1 0 0	1 0 0

Os 3 primeiros bits têm um significado especial:

- SUID - O Set User ID é representado pelo primeiro bit da sequência e afeta os ficheiros executáveis. O SUID permite que o ficheiro seja executado por outro qualquer utilizador, que não aquele que o criou, com as mesmas permissões do seu dono. Efetivamente, dá a um outro utilizador as mesmas permissões que o dono do ficheiro dispõe;
- SGID - O Set Group ID é representado pelo segundo bit da sequência. Funciona da mesma forma que o SUID, mas para grupos de utilizadores, dando permissões temporárias para um utilizador executar um ficheiro com as mesmas permissões que o grupo;
- Sticky - O Sticky é representado pelo terceiro bit da sequência. Permite que os arquivos fiquem temporariamente na memória mesmo depois de fechados, atuando assim como um género de *cache*, acelerando a sua execução numa futura chamada. Quando é aplicado a diretorias, impede utilizadores que não o dono do ficheiro da diretoria ou o super-utilizador (*root*) de renomear ou apagar ficheiros dentro da mesma.

Os seguintes 9 bits da sequência representam a presença ou não de cada um dos tipos de permissões em cada uma das classes existentes: dono, grupo ou outro. A representação no formato de um bit a 1 representa a presença de tal permissão, consoante o terceto onde se encontra. No exemplo acima, a presença de um bit a 1 na quarta posição da sequência, primeira posição do segundo terceto, tem o significado de que o dono do ficheiro, onde este controlo de acesso é aplicado, tem a permissão para ler o mesmo. Existem 8 possíveis combinações de permissões para cada uma das classes (tercetos), totalizando 512 arranjos possíveis de permutações para o controlo de acesso de um dado ficheiro ou diretoria.

Binário	Tradução	Permissões concedidas
000	- - -	Sem permissões
001	- - x	Execução
010	- w -	Escrita
011	- w x	Escrita e Execução
100	r - -	Leitura
101	r - x	Leitura e Execução
110	r w -	Leitura e Escrita
111	r w x	Leitura, Escrita e Execução

Para além das permissões até aqui abordadas (*read*, *write*, *execute*), que fazem parte do mecanismo tradicional, existem ainda outras duas permissões especiais, *s* e *t* [1], que se tratam de extensões ao mecanismo tradicional. Estas duas permissões dizem respeito à execução de ficheiros ou diretorias.

- *s* - Dependendo do local onde é usado: no terceto/secção do Dono, no terceto/secção do Grupo ou em ambos, indica o uso de um SUID, de um SGID ou de ambos. O uso de um s maiúsculo, *S*, indica a inexistência da permissão para execução;
- *t* - Aparece normalmente no terceto/secção Outro, e pretende indicar a existência de um *sticky bit*.

Alteração de privilégios

Naturalmente, existem mecanismos para proceder à alteração de privilégios de modo a conceder ou retirar os mesmos. O comando *chmod* permite proceder a tais mudanças [1, 6]. O comando *chmod* recebe como primeiro argumento a conversão para base 10 de cada um tercetos, consoante as permissões que queremos atribuir a cada uma das secções. Recebe como segundo argumento o nome do ficheiro/diretoria onde queremos aplicar as alterações. Por defeito, o utilizador que cria o ficheiro é, naturalmente, o seu dono e o grupo de utilizadores é aquele onde o dono se insere.

Consideremos o exemplo acima usado:

	Dono	Grupo	Outro
0 0 0	1 1 1	1 0 0	1 0 0
- - -	r w x	r - -	r - -

Como podemos verificar, após uma análise a este controlo de acesso (associado, por exemplo, a um ficheiro *teste.txt*), podemos concluir que o dono tem permissão para ler, escrever e executar o ficheiro. O grupo de utilizadores tem permissão apenas para o ler, assim como os outros utilizadores. Imaginemos agora o cenário onde surge a necessidade de atribuir todas as permissões possíveis, quer ao grupo de utilizadores, quer aos outros, à semelhança do que acontece com o dono.

Consideremos o exemplo acima usado. Queremos adicionar as permissões *write* e *execute*, às secções do Grupo e Outro.

Antes			
	Dono	Grupo	Outro
0 0 0	1 1 1	1 0 0	1 0 0
- - -	r w x	r - -	r - -
∑	7	4	4
Depois			
0 0 0	1 1 1	1 1 1	1 1 1
- - -	r w x	r w x	r w x
∑	7	7	7

Tendo em conta o que já sabemos acerca do comando *chmod*, basta proceder à execução do comando na seguinte forma para atribuir os privilégios que faltavam, quer a Grupo, quer a Outro.

```
chmod 777 teste.txt
```

Imaginemos agora o cenário onde nos víamos obrigados a retirar, por algum motivo, permissões aos grupos e outros. Efetivamente regressando ao controlo de acesso anterior à execução do comando *chmod*. Podemos para tal usar uma outra sintaxe aceite pelo comando *chmod*. Desta feita, os argumentos recebidos pelo comando serão os caracteres que representam cada uma das secções e privilégios.

Secção	Privilégio	Operador
a - all	*r - read*	*+ - add*
u - owner	*w - write*	*- - remove*
g - group	*r - execute*	*= - define*
o - other		

O comando no seguinte formato, efetivamente, retrocederia os efeitos do comando *chmod 777 teste.txt*

```
chmod u=rw,g=r,o=r teste.txt
```

	Dono	Grupo	Outro
0 0 0	1 1 1	1 0 0	1 0 0
- - -	r w x	r - -	r - -

Outros comandos.

- *chown* - Altera o dono do ficheiro;
- *chgrp* - Altera o grupo de utilizadores de um ficheiro.

Vulnerabilidades

Naturalmente, a existência deste mecanismo para proceder ao controlo do que os utilizadores têm efetivamente permissão para fazer, não é infalível.

Existem vulnerabilidades que podem ser exploradas, e uma rápida consulta à *National Vulnerability Database* [4] permite a consulta de algumas das mais graves.

Essencialmente, as vulnerabilidades que surgem nesta área enquadram-se todas na categoria da elevação de privilégios. Visto que são os privilégios que definem ou controlam o acesso e liberdade de ações em determinado ficheiro/diretoria, a grande preocupação para quem quer cometer algo ilegítimo e prejudicial será adicionar o máximo de privilégios possível, garantindo deste modo a maior liberdade possível.

CVE-2013-2171. Trata-se de uma vulnerabilidade que explora o ficheiro *sys/vm/vm_map.c*, de modo a abusar da função *vm_map_lookup* e essencialmente tirar partido do facto da mesma ter dificuldades em definir ou não o privilégio de escrita na memória, por parte de um determinado processo. Isto permite ao prevaricador executar uma elevação de privilégios para ele próprio poder executar operações de escrita à vontade.

CVE-2010-1146. Trata-se de uma vulnerabilidade que permite aos prevaricadores explorar o facto de não ser restringido o acesso, quer de escrita quer de leitura, da diretoria *.reiserfs_priv*. Aos malfeitores, basta eliminar o ficheiro *.reiserfs_priv/xattrs/*, e a partir deste momento passam a ter a capacidade de modificar atributos e ACLs, elevando assim os seus privilégios/permissões.

Conclusão

Com o conteúdo aqui exposto conseguimos ganhar uma maior perspetiva dos sistema de segurança implementados em SO modernos, nomeadamente Linux, que protegem o nossos sistemas de ataques por terceiros, e até mesmo de nós próprios, com as permissões existentes em ficheiros críticos do SO que nos impedem a sua modificação.

Temos, no entanto, noção que este ensaio representa uma visão algo superficial do sistema de controlo de acesso a Linux, mas considerámos que temos aqui uma base sólida, de onde qualquer leitor com uma maior curiosidade sobre o assunto poderá partir. Considerámos que o livro aqui mencionado [3] fornece um bom ponto de partida para quem quiser "mergulhar"mais a fundo no assunto, assim como todas as referências mencionadas.

Referências

[1] Luiz Arthur. *Sistemas Operacionais - Gnu/Linux Permissões de Arquivos Diretórios*. URL: https://pt.slideshare.net/luiz_arthur/sistemas-operacionais-gnulinux-permisses-de-arquivos-diretrios.

[2] *Gerenciamento de usuários e grupos do linux*. URL: https://pt.wikipedia.org/wiki/Gerenciamento_de_usu%C3%A1rios_e_grupos_do_linux.

[3] Dieter Gollmann. *Computer Security*. Wiley e Sons, Ltd., 2011.

[4] *National Vulnerability Database*. URL: https://nvd.nist.gov/vuln.

[5] Pedro Pinto. *Comandos Linux para Totós – Tutorial nº9*. URL: https://pplware.sapo.pt/tutoriais/comandos-linux-para-totos-%E2%80%93-tutorial-n%C2%BA9/.

[6] Pedro Pinto. *LINUX – Permissões em ficheiros e directórios*. URL: https://pplware.sapo.pt/linux/linux-permissoes-em-ficheiros-e-directorios/.

[7] *Utilizador e Superutilizador (root)*. URL: http://linux.caixamagica.pt/pag/documentacao/ManualUtilizador/HTML/node8.html.