unesp UNIVERSIDADE ESTADUAL PAULISTA

Administração de Redes TCP/IP

Tópico:
Conceitos de Segurança
Parte 3
Algumas Ferramentas e Técnicas Úteis

Prof. Dr. Adriano Mauro Cansian
adriano@dcce.ibilce.unesp.br
UNESP - IBILCE - São José do Rio Preto

Conceitos de Segurança[1]

Parte 3 - Algumas Ferramentas e Técnicas Úteis

Prof. Dr. Adriano Mauro Cansian
adriano@dcce.ibilce.unesp.br
UNESP - IBILCE - São José do Rio Preto

1. Ssh - Secure Shell [2]

1.1. O Que é o ssh

O ssh (*Secure Shell*) é um pacote de programas que permitem estabelecer uma conexão remota e também executar comandos remotamente.

Foi idealizado para substituir o rlogin, rsh (remsh) e o rcp e **prover um canal seguro (criptografado) entre duas máquinas**, através de um meio de transporte inseguro, e **sem a necessidade de estabelecer qualquer tipo de confiança entre os hosts**.

Pode-se ainda estabelecer conexões X Window criptografadas e da mesma forma forçar qualquer serviço TCP (através do redirecionamento de portas) a trafegar de forma cifrada.

1.2. Como funciona o ssh

Uma descrição detalhada do protocolo utilizado pelo ssh encontra-se em:

http://www.cs.hut.fi/ssh/

Há uma proposta de RFC em discussão disponível na forma de draft [2] especificando o protocolo utilizado pelo ssh.

---

[1] Créditos: este material foi produzido a partir de **"Tópicos em Segurança de Redes"** - Volume I, Versão 0.5.1 , produzido/traduzido/editado por Pedro A. M. Vazquez e outros (27/julho/98). Reproduzido sob permissão.

[2] Original de Nelson Murilo <nelson@pangeia.com.br>; publicado em **"Tópicos em Segurança de Redes"** - Volume I, Versão 0.5.1 Reproduzido sob permissão de Pedro A.M. Vazquez <*vazquez@iqm.unicamp.br*>.

1.3. Obtendo o ssh

O site oficial de distribuição do ssh é

ftp://ftp.cs.hut.fi/pub/ssh/

e em http://www.cs.hut.fi/ssh/ você encontra material abundante sobre o ssh e a localização de diversos mirrors internacionais.

O ssh roda basicamente em sistemas UNIX mas existem versões para MS-Windows feitas por Cedomir Igaly [refsshwin].

As distribuições oficiais do ssh para UNIX são assinadas com PGP usando o seguinte key ID:

DCB9AE01 1995/04/24 Ssh distribution key <ylo@cs.hut.fi>
Key fingerprint = C8 90 C8 5A 08 F0 F5 FD 61 AF E6 FF CF D4 29 D9

1.4. Instalação

Você necessita do perl para a compilação e instalação automática do textsfssh e para a execução de alguns scripts de manutenção que o acompanham.

Se você não possui o perl instalado ainda será possível compilar o ssh mas tornará as coisas mais difíceis. Desempacote o ssh com, por exemplo:

```
gzip -dc ssh-1.2.21.tar.gz | tar xvf -
```

A seguir mude para o diretório ssh-1.2.21, dê uma lida no arquivo INSTALL. Via de regra você só necessitará digitar:

```
# cd  ssh-1.2.21
# ./configure
# make
# make install
```

Isso irá gerar e instalar os dois programas, o ssh é o cliente que você utilizará para logar em máquinas remotas.

O sshd **é o server que executará no seu computador** se você desejar fornecer acesso via sessões de login criptografadas para os seus usuários.

Também serão criados os arquivos:

```
-rw-r--r--  1 root  wheel  855 Feb 11  1996 /etc/ssh_config
-rw-------  1 root  wheel  542 Feb 11  1996 /etc/ssh_host_key
-rw-r--r--  1 root  wheel  345 Feb 11  1996 /etc/ssh_host_key.pub
-rw-------  1 root  wheel  512 Sep 28 22:39 /etc/ssh_random_seed
-rw-r--r--  1 root  wheel  606 Mar 30  1996 /etc/sshd_config
```

E os seguintes comandos:

```
-rwxr-xr-x  1 root  bin    20172 Sep 27 16:15 make-ssh-known-hosts*
-rwxr-xr-x  1 root  bin    62621 Sep 27 16:15 scp*
lrwxrwxrwx  1 root  bin        3 Sep 27 16:15 slogin
lrwxrwxrwx  1 root  bin  1019857 Sep 27 16:15 ssh
lrwxrwxrwx  1 root  bin  1096208 Sep 27 16:15 sshd
-rwxr-xr-x  1 root  bin   603145 Sep 27 16:15 ssh-add*
-rwxr-xr-x  1 root  bin   605351 Sep 27 16:15 ssh-agent*
-rwxr-xr-x  1 root  bin    66474 Sep 27 16:15 ssh-askpass*
-rwxr-xr-x  1 root  bin   580955 Sep 27 16:15 ssh-keygen*
```

Você pode instalar apenas o ssh sem ser root, ele servirá para você logar em outros sistemas que rodem o sshd.

Você também pode instalar o sshd sem ser root. Para isso inicialize-o com a opção -p de forma que ele use uma porta não privilegiada (> 1024).

Isso permitirá que você realize conexões a partir de outras máquinas utilizando ssh -p mas apenas para a sua própria conta.

1.4.1. scp

Além dos programas ssh e sshd o pacote instala também um **substituto do comando rcp** denominado scp.

Este programa **realiza cópias de arquivos e diretórios para um computador remoto utilizando a mesma sintaxe do rcp** e utilizando os mesmos mecanismos de autenticação e criptografia do ssh.

1.5. Autenticação

Existem vários mecanismos que podem ser utilizados para autenticação de máquinas e usuários, que podem se conectar e/ou executar um comando remotamente.

Uma vez que a máquina origem da conexão tem permissão para efetuar alguma tarefa na máquina destino, um desses métodos deve ser usado.

1.5.1. Autenticação via hostname

Se a máquina origem está listada em **/etc/hosts.equiv** ou **/etc/shosts.equiv** na máquina destino, e **os usuários tem o mesmo nome em ambas as máquinas ou o usuário origem informar como argumento o nome do usuário destino imediatamente terá permissão para se logar**.

Alternativamente se **.rhosts ou .srhosts existem no diretório do usuário destino** e o **usuário tem nome idêntico em ambas as máquinas** ou o usuário origem informar como argumento o nome do usuário destino, **imediatamente terá permissão para logar**.

Note-se que **trata-se de equivalência e confiança pura e simples entre hosts, portanto esta forma de autenticação é normalmente desabilitada por ser insegura**.

É importante notar que **a insegurança refere-se a forma de validar o usuário e a máquina, e NÃO ao canal criptografado que será criado após a autenticação**.

1.5.2. Autenticação RSA

O ssh pode ter sua segurança melhorada enormemente, se forem acrescentados **algoritmos RSA baseadas na autenticação dos hosts**, passando a fazer uso também dos arquivos **/etc/ssh_known_hosts** e **$HOME/.ssh/known_hosts** e fazendo a **autenticação através de troca de chaves públicas**.

A cada login o ssh confere a identidade do host remoto com a informação contida no **ssh_known_hosts**, se ele achar uma inconsistência o usuário é alertado:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: 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 the host key has just been changed.
Please contact your system administrator.
Add correct host key in $HOME/.ssh/known_hosts to get rid of this message.
Agent forwarding is disabled to avoid attacks by corrupted servers.
X11 forwarding is disabled to avoid attacks by corrupted servers.
Are you sure you want to continue connecting (yes/no)?
```

Este método funciona com base no conceito de **chaves assimétricas ou pública/privada**.

A idéia é que cada usuário crie **seu par de chaves pública privada para propósitos de autenticação**.

A chave pública é mantida no diretório **$HOME/.ssh/authorized_keys**.

Quando o usuário estabelece a conexão as chaves em **authorized_keys** são consultadas para permitir, ou não, a efetivação do login.

O servidor verifica se a chave é permitida e, e somente se, envia um desafio (um número aleatório criptografado com a chave pública recebida).

A idéia de segurança reside no fato que **somente o usuário com a chave privada correta poderá descriptografar o desafio**, e portanto estará unicamente identificado.

O **ssh** implementa o protocolo de autenticação RSA automaticamente. O usuário pode criar seu par de chaves rodando o programa **ssh-keygen**.

A chave privativa é criada em **.ssh/identity** e a chave pública em **.ssh/identity.pub** ambas no diretório **$HOME** do usuário.

O arquivo contendo a chave pública (**identity.pub**) deve ser copiado com o nome de **authorized_keys** para o seu diretório **$HOME** na máquina destino.

Note-se que o arquivo **authorized_keys** corresponde ao **.rhosts** convencional e tem o formato de uma chave por linha.

Depois disso o usuário pode logar-se sem ter que informar a senha. Desnecessário falar que **a chave privada deve ser mantida sob vigilância constante, pois é o único elemento desconhecido em todo o processo**.

1.5.3. Autenticação via authsrv

Outro método de autenticação suportado pelo ssh utiliza o **authsrv**, **um servidor de autenticaçã**o que é distribuído pela TIS (*Trusted Information Services*).

Visto que os usuários do sistema não são necessariamente os usuários cadastrados no authsrv, a autenticação destes últimos pode ser feita **através de cartões como smartcard ou digipass**.

Neste caso **o nome do usuário é normalmente conhecido assim como o número de série do cartão autenticador**, o arquivo **/etc/sshd_tis.map** contém a equivalência entre o nome local e o nome correspondente na base de dados do TIS authserv.

Se o arquivo ou o usuário não existirem é assumido que o nome dado é sempre o nome local.

Se qualquer método acima falhar, o usuário será obrigado a entrar com a sua senha, então esta senha será repassada à máquina destino para verificação, sendo que todas estas informações (incluindo a senha) trafegarão criptografadas e, portanto, nenhum dado poderá ser capturado caso alguém esteja escutando a rede.

1.6. Que tipos de ataques o ssh protege?

Ataques baseados em escuta da rede (***sniffing***) **tornam-se ineficazes** neste caso, visto que todas as informações trafegam criptografas. Desta forma os dados podem ser lidos e copiados, mas não compreendidos.

Os algoritmos [3] utilizados pelo ssh para cifrar as informações são, atualmente IDEA (usado preferencialmente), DES (usado se IDEA não for suportado por ambos os hosts), DES, Blowfish, Arc-four, TSS ou *none* (i.e. nenhum, usado apenas para depuração).

A autenticação baseada simplesmente em equivalência e confiança de hosts é vulnerável a ataques como falsificação de número IP e nome de host.

Como a identificação do host é feita em função de uma informação que com mais ou menos habilidade pode ser falsificada, qualquer um que tenha conhecimento de quais máquinas confiam em um dado host, ou que de alguma forma conseguir incluir uma determinada máquina na lista de máquinas confiáveis tem plenas possibilidades de abrir uma sessão e/ou executar um comando remotamente.

A idéia do mecanismo de autenticação usado pelo ssh é minimizar as possibilidades descritas acima, por meio da criação de pares de chaves públicas/privadas, tanto para o host quanto para seus usuários.

Desta forma **além do host origem existir na lista de hosts permitidos do host destino**, ele ainda terá que responder a um desafio, ou seja, **descriptografar um número aleatório que foi cifrado com a chave pública informada pelo host origem**. Resposta essa que só o detentor da chave privada terá condições de fazer corretamente.

1.7. Que tipos de ataque o ssh não protege?

O ssh não oferece proteção **se o seu computador for invadido**, se o invasor obtiver acesso root na sua máquina ele pode corromper o ssh.

Se alguém obtiver acesso ao seu $HOMEDIR toda segurança fornecida pelo ssh será ineficaz; isto pode ocorrer se o seu $HOMEDIR for exportado via NFS.

Além da possibilidade de quebra do algoritmo de criptografia utilizado (considerando a qualidade dos algoritmos e tamanho das chaves essa possibilidade é pequena), o roubo da chave privada de um usuário e do host, aliada à falsificação de IP e/ou nome do host torna-se a única maneira de burlar os métodos de autenticação do ssh.

1.8. Redirecionamento de Serviços

O ssh **permite estabelecer um canal seguro (criptografado) para praticamente qualquer serviço TCP/IP**, isso pode ser feito em linha de comando ou através dos arquivos de configuração.

A principal vantagem desse método é permitir **que aplicações que não foram escritas com preocupação com segurança e privacidade** de dados (telnet, ftp, pop3, etc) **possam utilizar um canal seguro**.

O mecanismo utilizado para isso consiste em estabelecer uma conexão, efetuar todo o procedimento de autenticação baseado em chaves e em seguida estabelecer um canal entre a porta origem e destino do serviço escolhido.

Desta forma é estabelecido um canal seguro para trafegar os dados da aplicação sem que seja necessária nenhuma modificação no cliente ou no servidor original da aplicação e de maneira completamente transparente.

1.9. Exemplo: Transação POP3 segura

Suponha o redirecionamento do **pop3** entre os conhecidos hosts:

master : pop3d/110 e sshd/22
flood : pop3/cliente e ssh

Em **flood** digita-se:

```
% ssh -L 1100:master:110 master
```

Após a autenticação feita por **master**, **flood** estará escutando a porta 1100 (número de porta arbitrário, basta estar livre) e **estabelecido um canal seguro** entre flood e master conforme ilustrado abaixo:

Agora basta flood **direcionar o pop3 para a porta LOCAL 1100** e fazer a autenticação (do pop3) normalmente, que tudo estará sendo redirecionado para a porta 110 em master.

1.10. Exemplo: Redirecionamento de display X11

O estabelecimento de um canal seguro no caso do ambiente gráfico X Window é extremamente simples. Vamos aos passos:

master → sshd e aplicações X11 disponíveis no disco.
flood → ssh e servidor X11 rodando.

1. Inclua ou altere a seguinte linha no arquivo **/etc/sshd_config** em master:

```
X11Forwarding yes
```

2. Dispare ou redispare o sshd em master.
3. Em flood abra uma sessão em master (nenhuma opção é necessária visto que redirecionar X11 já é o default)

```
$ ssh master
```

Após a autenticação o ssh automaticamente atribuirá à variável DISPLAY o valor **master:n**, onde n é um inteiro e indica o canal por onde passará o protocolo X11 criptografado.

Para verificar este valor use:

Bourne Shell e compatíveis → echo $DISPLAY
Cshell → printenv $DISPLAY

Se você utiliza xdm o ssh utiliza o comando xauth quando faz a redireção de display, portanto este comando deve estar instalado no cliente e no path do usuário.

1.11. Referências para ssh

1. http://public.srce.hr/ cigaly/ssh ou ftp://hotline.pvt.net/pub/win_utils/winsock/ssh/
2. ftp://ftp.ietf.org/internet-drafts/draft-ietf-tls-ssh-00.txt
3. Schneier, B., Applied Cryptography: Protocols, Algorithms and Source Code in C, John Wiley, 1995, 2nd Ed., EUA.

2. MD5: assinaturas digitais

2.1. Introdução

O MD5 é um **algoritmo de assinatura digital** desenvolvido por Ron Rivest [1]. Acredita-se que **este algoritmo é irreversível** e que **é computacionalmente inviável produzir dois arquivos que possuam a mesma assinatura MD5** [2]. Isto torna este algoritmo ideal para a verificação da integridade de arquivos e executáveis.

2.2. Obtendo o MD5

O MD5 pode ser obtido em: **ftp://ftp.cert.org/pub/tools/md5**

O diretório contém o programa em C e documentação.

2.3. Compilando MD5

A compilação e instalação do md5 é direta, basta digitar **make** e copiar o executável para o diretório onde residirá.

Se o seu sistema operacional utiliza bibliotecas dinâmicas é aconselhável que você modifique o *Makefile* e adicione um flag para produzir um executável estático. Isto previne que um invasor consiga burlar a sua cópia do md5 através de corrupção de bibliotecas dinâmicas. Da mesma forma, é adequado que você gere a assinatura do executável compilado e a armazene em lugar seguro juntamente com uma cópia do programa.

2.4. Usando o MD5

A utilização do md5 é simples, para obter as assinaturas digitais de um arquivo digite md5 arquivo.

Exemplos:

```
% md5 /usr/bin/login
MD5 (/usr/bin/login) = 27713cf3152e7631257687cedc23848e

% md5 /usr/bin/netstat
MD5 (/usr/bin/netstat) = bd5a43cf3469fc6d9587397f2fd28098
```

Uma aplicação deste programa é na **geração de linhas de base do seu sistema operacional**. Isto é, **a geração de um arquivo contendo uma listagem de todos os programas e arquivos existentes logo após uma instalação a partir do zero**. Estas linhas de base, se armazenadas off-line, são extremamente úteis para a localização de programas adulterados com backdoors por invasores.

2.5. Referências sobre MD5

1. Rivest, R. RFC1321 The MD5 Message-Digest Algorithm, 1992.
2. Schneier, B., Applied Cryptography: Protocols, Algorithms and Source Code in C, John Wiley, 1995, 2nd Ed., EUA.

3. tripwire[1]

3.1. Introdução

Os arquivos e diretórios de um sistema normalmente são alvos de ataques. Os intrusos podem modificar bancos de dados do sistemas, alterar programas que gerenciam o sistema, remover logs do sistema, alterar configurações de daemons, alterar permissões de acesso, etc.

O **tripwire é uma ferramenta de teste de integridade** projetada para ambientes UNIX para **auxiliar o administrador do sistema a monitorar os filesystems com relação a alterações não autorizadas**. Foi desenvolvido por Gene Kim (gkim@cs.purdue.edu) e Eugene Spafford (spaf@cs.purdue.edu) da Universidade Purdue. Sua primeira versão foi disponibilizada em novembro de 1992.

As principais características são:

- Portabilidade: o código foi escrito em C (K & R) seguindo os padrões POSIX. Desse modo, **o programa pode ser compilado e instalado em praticamente todas as versões de UNIX** (BSD e System-V), incluindo Xenix e Unicos;

- As **bases de dados** geradas pelo tripwire **são arquivos texto** que podem ser lidas em um editor de texto comum (ou impressas);

- O tripwire inclui uma linguagem semelhante a linguagem M4, auxiliando o administrador a maximizar o re-uso dos arquivos de configuração;

- O tripwire distingue os arquivos de configuração e os arquivos de dados. Cada máquina pode compartilhar um arquivo de configuração, mas cada uma gera seu próprio arquivo de dados;

[1] Original de **Sidney Pio de Campos** <sidney@ifi.unicamp.br>; publicado em **"Tópicos em Segurança de Redes"** - Volume I, Versão 0.5.1 Reproduzido sob permissão do editor - Pedro A.M. Vazquez <*vazquez@iqm.unicamp.br*>.

- Possui uma interface genérica para rotinas de assinatura e suporta até 10 assinaturas para cada arquivo. As seguintes rotinas são incluídas na última distribuição:

  MD5
  MD4
  MD2
  4-pass Snefru
  128-bit HAVAL
  SHA
  CRC-32 e CRC-16

3.2. Obtendo o tripwire

A versão atual é a 1.2 que é descrita nesse texto. Ela pode ser obtida via ftp em:

ftp://coast.cs.purdue.edu/pub/COAST/Tripwire/tripwire-1.2.tar.Z
ftp://ftp.cert.org/pub/tools/tripwire/tripwire-1.2.tar.Z

3.3. Instalação

Após a transferência dos fontes e da descompressão (gzip e tar), devem ser seguidos os seguintes passos:

- Editar o Makefile de acordo com as ferramentas específicas do sistema (compilador, flags de compilação, etc);
- Escolher o arquivo correspondente ao seu sistema no diretório ./configs (conf-xxx.h, onde xxx corresponde ao seu sistema);
- Editar o arquivo ./include/config.h com o nome do arquivo conf-xxx.h do seu sistema;
- Alterar a localização dos arquivos citados em ./include/config.h de acordo com o seu sistema (variáveis ONFIG_PATH e DATABASE_PATH)
- Escolher o arquivo tw.config para seu sistema e transferir para o local indicado pela variável CONFIG_PATH. No diretório ./configs existem alguns exemplos de arquivos de configuração para alguns sistemas (SunOS, HPUX, Ultrix, BSD, Linux, etc). Caso não exista um arquivo de configuração para seu sistema, escolha o mais próximo possível e faça as alterações desejadas.
- No diretório base do tripwire (top-level) digite o comando make para construir o tripwire. A seguir execute make-test para verificar se tudo está correto.

3.4. O arquivo tw.config

O arquivo de configuração do tripwire (tw.config) contém uma lista de entrada, enumerando o conjunto de arquivos (ou diretórios) a serem verificados para mudanças, adições ou deleções. Associado a cada entrada existe uma máscara de seleção que descreve qual atributo do arquivo pode ser alterado sem que seja reportado. Cada linha do arquivo pode ser apresentada da seguinte forma:

[! |=] entrada [flags de seleção | template] [# comentário]

onde:

- **entrada →** nome do arquivo ou diretório. No caso de diretórios, a procura é recursiva descendente, mas sem cruzar filesystems. Os flags " ! " e " = " são utilizados para excluir da lista:

- **!** → Exclui inclusive a entrada: se for um arquivo, remove o arquivo da lista. Se for um diretório, o diretório e seus filhos serão removidos
- **#** → Exclui, exceto a entrada: se for um arquivo não tem efeito, mas se pode ser útil para diretórios que possuem arquivos temporários (por exemplo /tmp e /var/tmp)

- **flags de seleção →** descrevem os atributos dos arquivos ou dos inodes, tanto para incluir ou excluir os valores. A forma geral é do tipo

[[+ | -][pinugsam0123456789] ...]

onde:

-	ignora os atributos
+	registra de acordo com os atributos
p	permissão e bits dos modos do arquivo
i	número do inode
n	número de links
u	número do usuário dono do arquivo
g	número do grupo do dono do arquivo
s	tamanho do arquivo
a	timestamp do acesso ao arquivo
m	timestamp da modificação
c	timestamp da modificação/criação do inode
0,1,2,3,4,5,6,7,8,9	tipo de assinatura: null, MD5, Snefru, CRC-32, CRC-16, MD4, MD2, SHA, Haval, null (reservada para expansão)

Com o intuito de simplificar o uso, as seguintes abreviações podem ser utilizadas no campo flags de seleção:

R: Read-Only: (+pinugsm12-ac3456789) (default)
L: Log file (+pinug-sacm123456789)
N: ignorar Nada (+pinusgsamc123456789)
E: ignorar tudo (-pinusgsamc123456789)

A seguir apresentamos um exemplo de um arquivo tw.config:

```
# Arquivo exemplo tw.config
# arquivo/diretorio     mascara de selecao
/etc                R           # todos os arquivos embaixo de /etc
/etc/motd           L           # arquivos que podem mudar
=/var/tmp           R           # apenas o diretorio, nao os arquivos
```

3.5. Funcionamento do tripwire

Após a construção do tripwire, o primeiro passo é realizar a inicialização da base de dados.

A construção da base de dados é feita com o comando **tripwire-initialize**, que **lê o arquivo tw.config e gera as informações dos arquivos e diretórios apresentados**.

A base de dados gerada deve ser transferida para um local seguro: fita magnética ou disco seguro (a seguir apresentaremos um exemplo). Juntamente com a base de dados, deve-se transferir também o arquivo binário do tripwire e o arquivo de configuração (tw.config).

Uma vez construída a base de dados, **o comando tripwire pode ser utilizado para checar o estado dos arquivos listados**. Esse comando irá realizar a procura nos arquivos listados em tw.config e verificará se arquivos foram adicionados, deletados ou modificados.

Outra opção é utilizar o comando tripwire-interative. A cada alteração verificada o usuário é interrogado sobre a atualização na respectiva base de dados.

3.6. Exemplo de Uso

Na seção anterior descrevemos um uso geral do tripwire. Na prática devemos tomar alguns cuidados na utilização.

Uma possível configuração é inicialmente compilar, instalar e gerar a base de dados inicial em uma máquina em *single-user* (logicamente espera-se que essa máquina apresente todos os arquivos corretos)

- Escolhe-se um servidor seguro (de preferência onde poucas pessoas tem acesso pela rede).
- Transfere-se os arquivos para um disco exportado apenas para leitura (*read-only*) para as máquinas que devem ser monitoradas.
- Para realizar a verificação, pode-se colocar uma entrada no *crontab* da máquina a ser monitorada periodicamente, onde o disco read-only é montado, é executado o comando

tripwire, o resultado é enviado via e-mail para o administrador e posteriormente o disco é desmontado.

3.7. Considerações

O sucesso do uso do tripwire está diretamente relacionado ao arquivo de configuração utilizado e a observação de detalhes, como o armazenamento da base de dados em um local seguro. De nada adianta instalar o binário do tripwire ou a base de dados em um local onde possam ser facilmente alterados.

Cada assinatura pode ser incluída na máscara de seleção. O uso de cada tipo de assinatura vai depender da escolha do administrador que deve levar em conta a relação desempenho e segurança. Se a checagem estiver sendo muito demorada (ou com um processamento muito pesado) deve-se alterar os parâmetros do arquivo tw.config, diminuindo o número de assinaturas, por exemplo.

3.8. Referências sobre tripwire

1. http://www.cs.purdue.edu/coast/coast-tools.html
2. Kim, G.H., Spafford, E.H., The Design and Implementation of Tripwire: A File System Integrity Checker, Proceedings of the 2nd ACM Conference on Computer and Communications Security, 1994.

4. chkexploit - Detecção de Vulnerabilidades[1]

4.1. Introdução

Com a crescente popularização de diversos exploits tornou-se extremamente fácil o comprometimento de máquinas, tanto por usuários locais como remotos.

Embora muitos dos programas vulneráveis sejam prontamente corrigidos, **manter um sistema sempre atualizado nem sempre é tarefa fácil** para o administrador, principalmente em instalações antigas.

O chkexploit é uma ferramenta de segurança que implementa um conjunto de testes visando identificar versões de programas vulneráveis a exploits conhecidos.

[1] Original de **Klaus Steding-Jessen** <jessen@ahand.unicamp.br>; publicado em "Tópicos em Segurança de Redes" - Volume I, Versão 0.5.1 Reproduzido sob permissão do editor: Pedro A.M. Vazquez <*vazquez@iqm.unicamp.br*>.

É inteiramente escrito em *Bourne Shell* visando a maior portabilidade possível. Embora **seja mais específico para Linux e FreeBSD**, muitos dos seus testes são genéricos o suficiente para a maioria dos Unices. Nenhuma tentativa de explorar as vulnerabilidades é feita no processo de identificação das mesmas. **Toda a identificação dá-se apenas pelo exame das versões dos programas ou de seus arquivos de configuração**. Essa decisão foi tomada para que o chkexploit não pudesse ser usado como uma forma de ataque direto a essas vulnerabilidades. Nenhuma tentativa é feita também para a correção do problema, ficando de inteira responsabilidade do administrador fazê-lo. O chkexploit limita-se a relatá-lo, fornecendo uma breve descrição do problema e sua possível solução.

4.2. Instalação

4.2.1. Obtendo o chkexploit

chkexploit está disponível em:

ftp://ftp.pangeia.com.br/pub/seg/pac/chkexploit.tar.gz

Arquivo tar comprimido que é sempre a última versão do chkexploit.

ftp://ftp.pangeia.com.br/pub/seg/pac/chkexploit.tar.gz.md5

Assinatura MD5 do arquivo acima. Instruções de como verificar essa assinatura estão descritas a seguir.

4.2.2. Checando a Integridade do Pacote

É muito importante que a integridade do chkexploit seja verificada antes do seu uso, principalmente se não foi obtido do site principal.

Calcule a assinatura MD5 do arquivo que você acabou de receber e compare com a assinatura original disponível no site oficial.

```
 $ cat chkexploit.tar.gz.md5
28443ea67cab80acfdca13714e1dc01b  chkexploit-1.13.tar.gz

$ md5sum chkexploit.tar.gz
28443ea67cab80acfdca13714e1dc01b  chkexploit.tar.gz
```

Jamais use o chkexploit se a assinatura calculada não for igual à assinatura obtida do site oficial.

4.2.3. Obtendo Notificações de Novas Versões

Para receber notificação de novas versões do chkexploit mande mail para:

<chkexploit-announce-request@pangeia.com.br>

com subscribe chkexploit-announce no corpo da mensagem. Esta é uma lista de volume extremamente baixo, destinada apenas a anúncios de novas versões, não há discussão de segurança em geral. Para uma relação de tais listas consulte a seção 17.5.2.

4.3. Uso

O chkexploit pode ser executado sem argumentos, fazendo todos os testes, bem como especificando-se apenas os testes desejados na linha de comando.

A maioria dos testes não requer privilégios especiais para serem rodados. Alguns binários do sistema, entretanto, podem Ter apenas permissão de leitura ou execução para root exigindo do chkexploit privilégios para tal. Caso o script não tenha tais privilégios, um erro é relatado como resultado desse teste. Muitas instalações optam por deixar as permissões de execução e leitura de servers e binários suid apenas para root. Nesses casos, chkexploit terá dificuldades se não rodar sob este usuário.

Para cada teste efetuado, chkexploit pode apresentar uma breve descrição do problema e sua possível solução.

No caso de execução com sucesso do chkexploit, o total de vulnerabilidades detectadas é retornado como exit code do script.

4.4. Opções de Linha de Comando

Além do(s) nome(s) dos exploits que se deseja checar, as seguintes opções de linha de comando estão disponíveis:

- -h, --help
  Mostra as opções de linha de comando disponíveis e termina a execução.

- -v, --version
  Mostra a versão do programa e termina a execução.

- -q, --quiet
  Não mostra a descrição de cada exploit. Limita-se apenas ao resultado do teste.

- -V, --vulnerable
  Mostra apenas os testes vulneráveis. Essa opção implica no modo quiet.

- -l, --list
  Essa opção mostra a lista de exploits conhecidos e termina a execução.

- -d, --debug
  Faz chkexploit mostrar cada comando que está executando. Útil para relatar Bugs.

4.5. Exemplos do chkexploit

- Testa sendmail e imapd:

```
$ chkexploit sendmail imapd
sendmail: Not vulnerable
Problem: Version dependent.
Fix: Upgrade to latest version from ftp://ftp.sendmail.org.

imapd: Disabled
Problem: Non-local users may gain root or local access.
Fix: Disable imapd in /etc/inetd.conf or get the newest version.
```

- Testa apenas BIND, não mostrando a descrição do problema:

```
$ chkexploit --quiet bind
bind: VULNERABLE
```

- Mostra apenas as vulnerabilidades:

```
# chkexploit -V
suid_rw_partitions: VULNERABLE
ld_so: VULNERABLE
request_route: VULNERABLE
ssh: VULNERABLE
```

- Mostra o resultados de todos os testes, sem suas descrições, mandando mail para root:

```
# chkexploit -q | mail -s'chkexploit results' root
```

4.6. Limitações

- O número de exploits detectáveis ainda está muito aquém do desejado, principalmente para os Unices comerciais.

- A forma de adicionar novos testes ainda é editando o próprio programa. A forma ideal claramente seria a implementação de uma espécie de banco de dados de exploits, possivelmente separados por sistema operacional.

- chkexploit baseia-se principalmente no output dos programas a serem analisados como, por exemplo, versão ou alguma outra palavra característica. Nenhuma informação sobre a integridade do arquivo é levada em conta como, por exemplo, uma assinatura digital. É possível para um cracker, depois de comprometer uma máquina, editar essas informações nos arquivos de tal modo a não levantar suspeitas em testes futuros.

5. Referências Online

Devido à maneira muito dinâmica com que novas vulnerabilidades vão sendo descobertas é fundamental manter-se sempre atualizado. Segue uma lista de referências onde é possível obter anúncios sobre os últimos furos de segurança, novos exploits e como proteger-se desses ataques.

5.1. Links Sobre Vulnerabilidades

5.1.1. Boletins de Vulnerabilidades

- **http://www.cert.org**

Computer Emergency Response Team, boa fonte de boletins de segurança, ferramentas e artigos.

- **http://www.auscert.org.au**

Australian Computer Emergency Response Team, boa fonte de boletins de segurança, ferramentas e artigos.

- **http://ciac.llnl.gov**

Computer Incident Advisory Capability. Anúncios regulares sobre vulnerabilidades em diversos sistemas.

5.1.2. Exploits

- **http://oliver.efri.hr/~crv/security/bugs/list.html**

Security Bugware List Page. Excelente lista de exploits e vulnerabilidades, agrupados por sistema operacional. Atualizado com bastante freqüência.

- **http:/www.rootshell.com/**

O famoso *RootShell*. Extenso repositório de exploits para diversos sistemas.

- **http://www.dhp.com/~fyodor/sploits.html**

Fyodor's Exploit World, extenso compilação de exploits separados por sistema e categoria.

- **http://www.nmrc.org/files/blackhat.html**

Coleção de exploits e ferramentas de cracking, separados por sistema.

- **http://www.users.interport.net/~reptile/linux/index.html**

Reptile's Linux Security Page. Página sobre exploits específicos de Linux.

- **http://www.ecst.csuchico.edu/~jtmurphy/**

Linux Security Home Page. Vários exploits e dicas para tornar sistemas baseados em Linux mais seguros.

5.1.3. Listas de Discussão Internacionais e outros recursos

- **Lista Eletrônica Bugtraq <bugtraq@netspace.org>**

Excelente lista de discussão sobre vulnerabilidades e segurança em geral. Para inscrever-se, é necessário mandar mail para <listserv@netspace.org> e no corpo da mensagem subscribe bugtraq. Mas, atenção, esta lista de discussão tem grande volume de tráfego (em geral, mais de 30 mensagens ao dia).

Todas as mensagens desde Junho de 1995 podem ser consultadas em **http:/www.netspace.org/lsv-archive/bugtraq.html** ou em **http://www.geek-girl.com/bugtraq/archives.html**

- **Lista eletrônica 8lgm : http://www.8lgm.org/**

5.2. Relatando Bugs

Bugs e comentários devem ser enviados para:

<chkexploit-bugs@pangeia.com.br>

Ao relatar possíveis Bugs do chkexploit, veja se o problema enquadra-se num dos casos:

- Falso alarme: o chkexploit está relatando como vulnerável algo que você tem certeza que não é. Tente mandar, por exemplo, o resultado de um exploit que está falhando, mostrando que seu sistema não é vulnerável a esse ataque.

- Você tem certeza que seu sistema é vulnerável a este problema mas o chkexploit diz que não. Tente mandar algum resultado que confirme isso.

Tente ser o mais preciso possível, relatando tipo e versão do seu sistema operacional, versões de bibliotecas, patches aplicados e qualquer outra informação que achar relevante.

De preferência, mande a saída de:

```
chkexploit --debug nome_do_teste
```

Onde `nome_do_teste' é o nome do teste que você desconfia que está com problemas.

6. chkrootkit[1]

6.1. Introdução

Uma das tarefas do administrador de sistemas UNIX é poder determinar quais usuários utilizaram e quais estão utilizando o seu sistema. Para obter essas informações ele dispõe de uma série de utilitários, que indicarão os vários tipos de atividades, tais como: quais e quantos processos estão ativos, que uso está sendo feito da rede, carga da cpu, quantos e quais são os usuários ativos, entre outras. Sem contar as ferramentas (altamente recomendadas) dos subsistemas de auditoria e contabilidade.

Neste cenário, qualquer atividade hostil tanto local quanto remota que venha a ser reportada ao administrador este pode facilmente identificar o responsável pela ação. O administrador usaria alguns comandos tradicionais do UNIX para saber, por exemplo, quais usuários estavam logados no momento da atividade reportada e qual estava na porta identificada como atacante. Caso o ataque estivesse em andamento, ele poderia dispor de comandos para verificar quem está usando no momento e quem está usando a porta atacante, além dos comandos que este usuário está usando no momento.

Desta forma, um usuário com más intenções acharia útil ter uma forma de se ocultar dos olhares do administrador. Isso motivou o surgimento de vários programas que tentavam apagar rastros dos comandos e até mesmo da presença de usuários em um sistema UNIX. **Esses programas começaram a ser disseminados juntos, formando uma espécie de kit, tornando-se então famoso um conjunto de programas para máquinas Sun, que ganhou o nome de r00tkit** .

Com a popularização da Internet e o acesso a UNIX gratuitos como o Linux. O r00tkit não só foi portado para Linux como ganhou alguns recursos adicionais, estava pronto o Linux r00tkit. Nunca é demais lembrar que tecnicamente não existem maiores problemas de porte para outras plataformas como, por exemplo: Open/Net/FreeBSD, BSDI, SCO, AIX, HP-UX, etc.

6.2. Objetivos

O **r00tkit é um conjunto de ferramentas** que podem ser usadas em conjunto ou isoladamente, e não são, de forma alguma, interdependentes**. Isto dificulta a tarefa de identificação**, pois, pelo fato de trabalharem isoladamente **não reproduzem padrões facilmente identificáveis**. Some-se a isso o fato do pacote ser distribuído em formato fonte, o que permite alteração das informações default.

[1] Original de **Nelson Murilo** <nelson@pangeia.com.br>; publicado em **"Tópicos em Segurança de Redes"** - Volume I, Versão 0.5.1 Reproduzido sob permissão de Pedro A.M. Vazquez <*vazquez@iqm.unicamp.br*>.

A idéia do chkrootkit foi aliar a busca por essas informações default à verificação de determinados padrões que caracterizam algumas das ferramentas, afim de, com isso, identificar se uma determinada instalação está ou não comprometida.

6.3. Construção

O principal problema de programas que fazem verificação de algum tipo de vulnerabilidade é que eles costumam explorar a mesma, tornando o programa também uma ferramenta de ataque, e não apenas de prevenção. Portanto a idéia foi fazer um pacote que não tivesse esse efeito colateral de ataque, para isso foi utilizada a mesma idéia usada no chkexploit e está presente também em parte em pacotes como o Cops, que é tentar identificar determinadas cadeias nos arquivos executáveis, e (no caso específico do chkrootkit) padrões de comportamento de determinados programas.

A linguagem escolhida para o núcleo do pacote foi Bourne shell, visando elevar ao máximo a possibilidade do pacote rodar sem alterações nos mais variados ambientes, e permitir mudanças rápidas em situações específicas. A despeito disso existem pequenos programas em C, visto que praticamente todos os sistemas operacionais UNIX dispõe de um compilador C. O chkrootkit está disponível hoje para os seguintes sistemas: Linux e FreeBSD .

6.4. Uso

O pacote é composto de um Bourne shell script e três pequenos programas em C que, após compilados serão chamados a partir do script chkrootkit, sendo esse, portanto o programa principal.

Como não há necessidade de instalar o r00tkit completamente e nem alterar todos os programas que o compõe, o script verifica todos programas possíveis de serem alterados, bem como os utilitários para tarefas específicas, como escutar a interface Ethernet para adquirir senhas e apagar registros de entrada e saída.

Uma saída típica do chkrootkit seria:

```
# ./chkrootkit
Checking `chfn'...Not vulnerable
Checking `chsh'...Not vulnerable
Checking `inetd'...Not vulnerable
Checking `login'...Not vulnerable
Checking `ls'...Not vulnerable
Checking `du'...Not vulnerable
Checking `ifconfig'...Not vulnerable
Checking `netstat'...Not vulnerable
Checking `passwd'...Not vulnerable
Checking `ps'...Not vulnerable
Checking `top'...Not vulnerable
Checking `rshd'...Not vulnerable
Checking `syslogd'...Not vulnerable
```

```
Checking `bindshell'...Not vulnerable
Checking `z2'...Nothing deleted
Checking `wted'...Nothing deleted
Checking `sniffer'...
eth0 is not promisc
eth1 is not promisc
Checking `aliens'...No suspect files
```

6.5. Onde obter

O pacote esta disponível oficialmente em:

ftp://ftp.pangeia.com.br/pub/seg/pac/chkrootkit-0.8.tar.gz

7. ipfilter

7.1. Introdução

A filtragem de pacotes em um gateway ou host pode reforçar significativamente a segurança se associada a outras medidas de controle de acesso. Esta filtragem não necessita de um computador poderoso, podendo ser efetuada por uma máquina de dimensões modestas como um 486.

Existem várias opções comerciais de software para realizar este tipo de tarefa mas elas são onerosas. O Linux e o FreeBSD oferecem implementações (ipfwadm e ipfw) bastante poderosas mas restritas a estes sistemas operacionais. O pacote ip-filter desenvolvido por Darren Reed, por outro lado, suporta várias plataformas e permite a implementação de filtros de pacotes nos seguintes sistemas operacionais: BSD/OS, FreeBSD, Irix 6.2, OpenBSD, NetBSD, e Solaris 1 e 2.

O ip-filter é capaz de realizar as seguintes funções:

1. Permitir ou negar explicitamente o tráfego de qualquer pacote;
2. Distinguir pacotes de acordo com as interfaces do sistema;
3. Filtrar redes ou hosts;
4. Filtrar seletivamente qualquer protocolo IP;
5. Filtrar seletivamente pacotes IP fragmentados;
6. Filtrar seletivamente pacotes com opções IP;
7. Enviar códigos ICMP ou TCP RST em resposta a pacotes bloqueados;
8. Manter informações de estado para tráfego TCP, UDP e ICMP;
9. Manter informações de estado de fragmentos de qualquer pacote IP, aplicando a mesma regra a todos os demais fragmentos do pacote;
10. Operar como Network Address Translator (NAT);
11. Usar redirecionamento para montagem de conexões via proxy totalmente transparentes;

Para os três protocolos mais utilizados na Internet (TCP, UDP e ICMP)o ip-filter permite a filtragem utilizando os seguinte critérios:

- número da porta ou faixa de portas de pacotes TCP ou UDP;
- tipo ou código do pacote ICMP;
- conexões TCP estabelecidas;
- qualquer combinação de flags TCP;
- pacotes IP fragmentados com headers incompletos;
- qualquer uma das 19 opções ou das 8 classes registradas ICMP;
- campo Type of Service (TOS)

7.2. Obtendo o ip-filter

O ip-filter é distribuído gratuitamente sob uma licença estilo BSD e pode ser obtido via ftp [1] ou http a partir do seu site oficial na Austrália:

http://coombs.anu.edu.au/~avalon/ip-filter.html

A versão atual é a 3.1.11, a versão 3.2 beta com suporte para Irix já está disponível e é a escrita neste texto.

7.3. Instalação

O ip-filter pode ser instalado permanentemente no kernel ou utilizado como um *lkm* (*loadable kernel module*) para kernels baseados em 4.xBSD ou como um *lkm* para Irix e Solaris 2. Cada sistema possui uma opção correspondente no Makefile que compilará os componentes necessários do filtro e os seguintes utilitários:

- **ipf**
  gerenciador do ip-filter para adição ou remoção de regras de filtragem;
- **ipftest**
  programa de teste de regras de filtragem;
- **ipmon**
  monitoramento de logs;
- **ipnat**
  gerenciador do NAT para adição e remoção de regras de tradução de endereços de rede;
- **ipresend**
  programa para reenvio para a rede de pacotes capturados;
- **ipsend**
  programa para a geração e envio de qualquer tipo de pacote IP;
- **iptest**
  programa para testes e geração automática de pacotes IP;
- **mkfilters**
  script de geração automática de conjunto básico de regras de filtragem.

7.4. Algumas Considerações

O objetivo deste texto é de apenas apresentar o ip-filter. O ip-filter é uma ferramenta poderosa que pode tornar um computador de pequeno porte (ex. um 486) e fora de uso num gateway com capacidade de filtragem de pacotes. É importante ressaltar que um firewall construído unicamente utilizando esta técnica não é suficiente para proteger uma rede de um invasor bem determinado. Conforme ilustrado por Chapman [2] e Cheswick [3] um firewall seguro necessita a implementação de outras técnicas além da filtragem de pacotes.

7.5. Referências sobre o IP-FILTER

[1.]

Austrália:	ftp://coombs.anu.edu.au/pub/net/kernel/
Finlândia:	ftp://nic.funet.fi
Reino Unido:	ftp://ftp.tardis.ed.ac.uk
Grécia:	ftp://ftp.ntua.gr
EUA:	
	ftp://ftp.gw.com/pub/unix/ip-filter/
	ftp://ftp.zyzzyva.com
	ftp://ftp.umbc.edu
Japão:	ftp.win.or.jp

[2.] Chapman, D. B., Zwicky, E. D., Building Internet Firewalls, O'Reilly & Associates, Inc., EUA, 1995.
[3.] Cheswick, W. R., Bellovin, S. M., Firewalls and Internet Security, Addison Wesley, EUA, 1994.
[4.] Garfinkel, S., Spafford E., Pratical UNIX and Internet Security, 2nd ed., O'Reilly & Associates, Inc., EUA, 1996.

ANOTAÇÕES:

Arquivo: **net-sec3.pdf**
Última atualização: quinta-feira, 14 de novembro de 2002 19:12:38