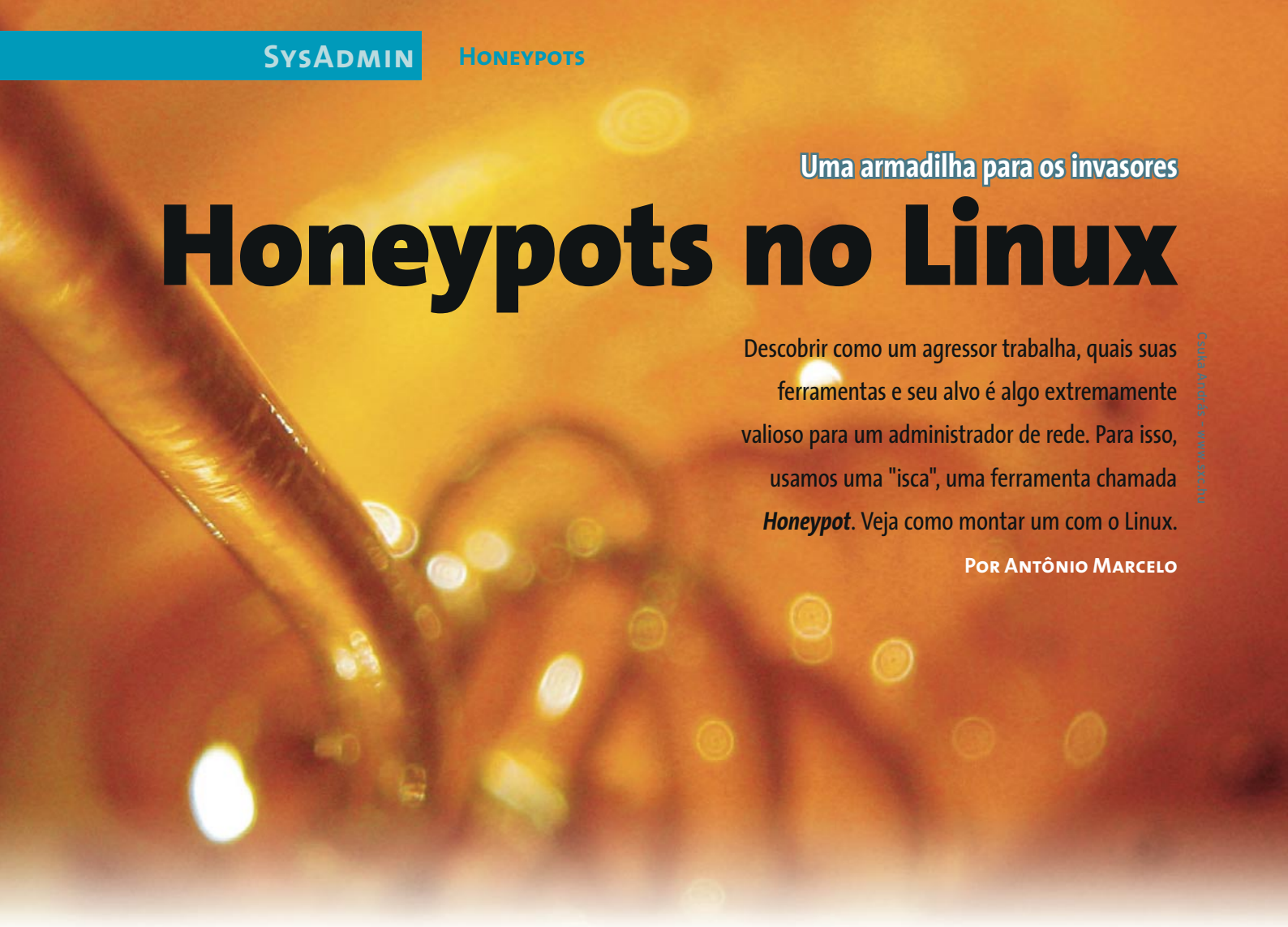

Uma armadilha para os invasores

Honeypots no Linux

Descobrir como um agressor trabalha, quais suas ferramentas e seu alvo é algo extremamente valioso para um administrador de rede. Para isso, usamos uma "isca", uma ferramenta chamada ***Honeypot***. Veja como montar um com o Linux.

Por Antônio Marcelo

Nos dias atuais, quando a segurança tornou-se uma preocupação constante dos administradores de rede, conhecer as ameaças que podem comprometer uma estrutura operacional e permitir o roubo de informações é uma necessidade. Muitos consideram-se protegidos por firewalls, sistemas de detecção de intrusão ou até mesmo por uma política de segurança implementada na empresa. Os cuidados tomados frente a diversas ameaças e situações concentram-se sobretudo na prevenção e na resposta rápida a um evento qualquer de quebra de segurança.

Contudo, uma nova idéia na área de segurança da informação vem surgindo e ganhando um número cada vez maior de adeptos: a do estudo dos agressores e de seus *modi operandi*. Trata-se de uma atividade que permite descobrir como um invasor mal-intencionado "trabalha" e ao mesmo tempo identificar suas ferramentas – e o mais importante – seu alvo. Essas técnicas de estudo "in vivo" devem ser utilizadas de maneira bastante cautelosa, já que muitos administradores acabam abrindo uma brecha em suas redes, em vez de criar um ambiente de estudo isolado.

O objetivo deste artigo é mostrar como pode ser realizada a montagem de um *Honeypot* em Linux, obter resultados interessantes em situações reais e traçar um perfil dos ataques a uma rede e de como seus agressores estão agindo. Essa atividade será realizada em uma máquina com a distribuição Slackware instalada e com alguns recursos de segurança ativos.

Os Honeypots

Existem definições clássicas como a de Lance Spitzner (veja referência [1]), criador do projeto Honeynet, que diz que:

"Um honeypot é um recurso de rede cuja função é ser atacado e comprometido (invadido). Significa dizer que um honeypot poderá ser testado, atacado e invadido. Os honeypots não fazem nenhum tipo de prevenção; eles fornecem informações adicionais de valor inestimável"

Outra definição seria a seguinte:

"Um honeypot é um sistema que possui falhas de segurança, reais ou virtuais, colocadas de maneira proposital, a fim de que seja invadido e o fruto dessa invasão possa ser estudado"

Muitos administradores pensam erroneamente que um honeypot é uma ferramenta de segurança. Engano: é uma ferramenta de estudo utilizado na segurança. Por meio de um honeypot é possível desenvolver toda uma metodologia e novas ferramentas para combater os agressores.

Os honeypots podem ser classificados de acordo com seus níveis de operação, que seriam os seguintes:

- Baixa interatividade: esse tipo de honeypot provê os chamados serviços falsos em determinadas portas TCP/UDP. No fundo, esse honeypot é um *listener* tcp/udp, aguardando conexões e devolvendo ao atacante respostas falsas;
- Média interatividade: nesse tipo existe um envolvimento maior, já que o honeypot irá simular, com muitos detalhes, um ambiente totalmente falso. Na realidade ele seria como uma espécie de "concha" – o atacante ficaria preso num ambiente controlado, sem contato com o sistema real. Os servidores executados respondem às requisições, inclusive simulando bugs e outras peculiaridades de comportamento;
- Alta interatividade: esse honeypot na realidade é uma máquina executando um sistema operacional real com serviços comprometidos e que estaria servindo como isca num ponto real da Internet. Nesse caso, o sistema possui uma série de recursos de monitoração e proteção. Normalmente fica localizado em uma rede isolada e com um *firewall* ou *bridge* na frente. Apesar dos enormes riscos, os honeypots de alta interatividade são bastante usados e tornam-se elementos muito importantes na captura de técnicas e informações sobre os agressores. O risco é a sua principal vantagem, já que atrai o invasor.

Colocaremos em produção um honeypot de média interatividade utilizando como configuração o seguinte:

- Uma máquina Linux (no nosso caso, rodando o Slackware com o kernel 2.4.29) com a rede configurada
- O software *Honeyperl* do projeto HoneypotBR
- O sistema de detecção de intrusão *snort*
- Um filtro de pacotes baseado no *iptables* com regras básicas para nossos serviços.

Iniciando

O processo de instalação do Slackware (ou sua distribuição favorita) deve ser feito respeitando-se os requisitos abaixo:

- Perl versão 5.6.0 ou superior;
- Nenhum servidor de rede instalado (ftp, web, correio etc);
- Nenhum serviço de rede sendo executado pelo *inetd*;
- Placa de rede configurada;
- Uma conexão ativa com a Internet (pode ser via ADSL);

Com isso, podemos começar a trabalhar com o snort.

Instalando o IDS

A idéia dos IDS (*Intrusion Detection Systems* – Sistemas de Detecção de Intrusão) surgiu no início da década de 80. Seu objetivo era desenvolver um sistema de auditoria que pudesse, além de registrar a invasão, modificar e preparar o sistema para resistir a ela e, se possível, lançar um contra-ataque. Um IDS tem as seguintes características:

- Análise e monitoração das atividades de usuários e serviços;
- Auditoria e levantamento de vulnerabilidades do sistema;
- Reconhecimento dos padrões de ataque mais comuns;
- Reconhecimento dos padrões de violação do sistema;
- Análises estatísticas de atividades incomuns ao sistema;
- Integração com a auditoria (logs) do sistema;
- Reativo a ataques e atividades incomuns (opcional);

Utilizaremos o snort (que pode ser obtido em [2]) como exemplo de IDS. Como nossa intenção não é fazer um manual de instalação do snort, seremos o mais breve possível. Baixe o código fonte do programa, descompacte-o e compile/instale com a velha seqüência `./configure; make; make install`

Listagem 1: regras do snort ativas

```
01 include $RULE_PATH/bad-traffic.rules
02 include $RULE_PATH/exploit.rules
03 include $RULE_PATH/scan.rules
04 include $RULE_PATH/finger.rules
05 include $RULE_PATH/ftp.rules
06 include $RULE_PATH/telnet.rules
07 include $RULE_PATH/rpc.rules
08 include $RULE_PATH/rservices.rules
09 include $RULE_PATH/dos.rules
10 include $RULE_PATH/ddos.rules
11 include $RULE_PATH/dns.rules
12 include $RULE_PATH/tftp.rules
13 include $RULE_PATH/web-cgi.rules
14 include $RULE_PATH/web-coldfusion.rules
15 include $RULE_PATH/web-iis.rules
16 include $RULE_PATH/web-frontpage.rules
17 include $RULE_PATH/web-misc.rules
18 include $RULE_PATH/web-client.rules
19 include $RULE_PATH/web-php.rules
20 include $RULE_PATH/sql.rules
21 include $RULE_PATH/x11.rules
22 include $RULE_PATH/icmp.rules
23 include $RULE_PATH/netbios.rules
24 include $RULE_PATH/misc.rules
25 include $RULE_PATH/attack-responses.rules
26 include $RULE_PATH/oracle.rules
27 include $RULE_PATH/mysql.rules
28 include $RULE_PATH/snmp.rules
29 include $RULE_PATH/smtp.rules
30 include $RULE_PATH/imap.rules
31 include $RULE_PATH/pop2.rules
32 include $RULE_PATH/pop3.rules
33 include $RULE_PATH/nntp.rules
34 include $RULE_PATH/other-ids.rules
35 include $RULE_PATH/web-attacks.rules
36 include $RULE_PATH/backdoor.rules
37 include $RULE_PATH/shellcode.rules
38 include $RULE_PATH/virus.rules
```

O próximo passo é configurar o snort. O primeiro passo é criar um diretório `snort` dentro do diretório `/etc`, com o comando `mkdir /etc/snort`. Dentro do diretório com o código fonte do snort existe o subdiretório `etc`, que contém os arquivos de configuração do programa. Copie-os para `/etc/snort`:

```
cd snort-2.0.x
cp etc/* /etc/snort
```

Feito isso, o próximo passo é copiar as regras de dentro do subdiretório `rules`, no diretório com o código fonte do snort, para o diretório `/etc/snort`. Essas regras contêm as assinaturas dos ataques feitos ao IDS. Copie-as com o comando a seguir: `cp -r rules/ /etc/snort`.

Configuração do snort

A configuração do snort é feita através do arquivo `snort.conf`, em `/etc/snort`. Para ajustá-lo às necessidades de nosso honeypot, edite o arquivo de acordo com os parâmetros da **listagem 2**.

As demais regras que definem assinaturas e strings têm de ser escolhidas ao gosto do freguês e não serão citadas neste artigo. Agora devemos escolher as regras de detecção a utilizar. Basta descomentar a linha correspondente, retirando o caracter # do início:

```
include $RULE_PATH/regra_a_usar.rules
```

Cada uma das regras cobre ataques específicos. Recomendamos a leitura do manual do snort para maiores esclarecimentos. Na **listagem 1** temos a lista das regras ativas em nosso caso. tDessa forma o snort estará pronto para uso.

Configurando o Honeyperl

O *Honeyperl* é um software de fácil instalação e configuração desenvolvido em Perl. Pode ser obtido no site do projeto HoneypotBR em [3] e instalado de maneira bem rápida. Trata-se de um honeypot de média interatividade que simula os seguintes serviços: Squid, Apache, servidores de e-mail (Sendmail, Postfix, Qmail e MSExchange), FTP, Echo e POP3.

Ele ainda é capaz de capturar assinaturas de vírus num pequeno *Tarpit* (poço de piche) implementado. A única dependência é que o PERL (versão 5.6.0 ou superior) esteja instalado no sistema. Para configurar o Honeyperl basta executar a seguinte seqüência de comandos:

- Descompacte o pacote com o comando `tar -xzvf honeyperl.0.0.7.1.tar.gz`
- Execute o comando `perl verify.pl` para verificar se todos os módulos Perl necessários para a execução do programa. Caso seja necessário, essa rotina acessa automaticamente o repositório CPAN e executa o download e a instalação dos módulos Perl faltantes.

O processo de configuração deve ser feito de forma bastante cuidadosa. Inicialmente devem ser escolhidos os serviços que serão executados no Honeypot. O Honeyperl fornece dois tipos de arquivo de configuração:

`honeyperl.conf`: Arquivo principal de configuração do programa. Fica localizado no diretório `conf` juntamente com o dos outros módulos. Dentro dele há algumas variáveis importantes que devem ser configuradas. São as seguintes:

```
##############
#Secao 1     #
##############
```

Listagem 2: snort.conf

```
01 var HOME_NET $ppp0_ADDRESS
02 #ppp0 representa uma conexão discada à Internet
03 var EXTERNAL_NET any
04 #define a rede externa de qualquer ip, ou seja, qualquer um
05 var DNS_SERVERS $HOME_NET
06 #define os serviços cujo tráfego será analisado pelo snort
07 var SMTP_SERVERS $HOME_NET
08 #define o servidor SMTP
09 var TELNET_SERVERS $HOME_NET
10 #define o servidor Telnet
11 var HTTP_PORTS 80
12 #define a porta do servidor HTTP
13 var SHELLCODE_PORTS !80
14 #define qualquer porta diferente (!) de 80
15 var RULE_PATH ./rules
16 #define o diretório das regras como ./rules
```

```
#Dominio que será utilizado pelos fakes
dominio=honeypot.com.br
```

Essa variável permite a definição de um nome de domínio. Coloque o da sua instituição/empresa. Esse parâmetro faz com que, na hora em que o intruso fizer o levantamento de informações sobre o servidor, receba como resposta o nome definido. Exemplo: `servidor.honeypot.com.br`

```
#email utilizado nos fakes
email=admin@honeypot.com.br
```

Um endereço de email de sua escolha para aparecer como contato nas respostas falsas que os serviços irão emitir. Recomendamos um endereço falso ou uma conta especialmente selecionada para isso.

```
#Usuário utilizado
usuario=root
```

Usuário para execução do Honeyperl. Deixe-o como root.

```
#Deseja ver as mensagens no terminal?
#opcoes:(sim/yes)/(nao/no)
terminal=sim
```

Mostra as mensagens referentes a ataques em tempo real no terminal, permitindo assim seu acompanhamento.

```
#Deseja ativar firewall
#opcoes:(sim/yes)/(nao/no)
firewall=nao
```

Essa opção tem que ser manipulada com muito cuidado. Estando ativa, o Honeyperl irá gerar um arquivo de regras de firewall baseado no iptables, no sub-diretório `firewall` e atualizará a tabela do iptables. Alguns usuários tiveram problemas com bloqueio de endereços. Recomendamos não ativar essa opção e, em seu lugar, utilizar o script no final do artigo.

```
#Sistemas de firewall disponíveis.
#Pode-se ter linux22, linu24 ou openbsd:
#openbsd : trabalha com PF
```

Listagem 3: iptables com mel

```
01 #! /bin/sh
02 #
03 # rc.firewall
04 #
05 # Por Antonio Marcelo - amarcelo@plebe.com.br
06 #Script padrão para firewalls baseados em iptables
07 #
08 #
09
10 if [ "$1" = "flush" ]; then
11    echo "Flushing"
12    iptables -P INPUT ACCEPT
13    iptables -P FORWARD ACCEPT
14    iptables -P OUTPUT ACCEPT
15    iptables -F
16    iptables -t nat -F   # Flush no NAT
17          iptables -X    # Flush nas CHAINS PERSONALIZADAS
18          iptables -Z # Zera regras especificas. Qdo nao houver
19             # argumentos, zera todas as regras. Idem ao -f.
20    echo "done"
21 else
22 echo "Iniciando firewall"
23
24 iptables -P INPUT DROP
25 iptables -P FORWARD DROP
26 iptables -P OUTPUT ACCEPT
27
28 #Liberando as portas de nosso honeypot
29 iptables -A INPUT -p tcp -m multiport --destination-port 20,21,25,80,110,3128
30 iptables -A INPUT -p TCP -m state --state ESTABLISHED,RELATED -j ACCEPT
31
32 FI
```

```
#linux24 : IPTables
#linux22 : ipchains
so=linux24
```

Com essa opção, podemos definir se o firewall vai ser feito usando o *PF*, do OpenBSD, o *ipchains* ou o *iptables*.

A segunda seção do arquivo de configuração é que determina os serviços falsos (os *fakes*) a serem executados. Dependendo da simulação desejada podemos ativar ou não determinado fake.

```
##############
#Secao 2    #
##############

#Fakes a serem iniciados
fakesquid:squid:conf/fakesquid.conf:3128:Squid Emul
```

```
fakesmtp:smtp:conf/fakesmtp.conf:25:SMTP emul
fakehttpd:httpd:conf/httpd.conf:80:HTTPD emul
fakepop3:pop3:conf/pop3:110:POP3 emul
#fakeecho:echo::7:Echo emul
fakeftp:ftp:conf/fakeftp.conf:21:FTP emul
#fakepit:pit:20001:Pit emul
```

A estrutura da definição de um fake é a seguinte:

```
nomedofake:modulo(servico):arquivo de config:porta TCP:comentario
```

Colocar uma cerquilha (#) na frente de qualquer um dos parâmetros impede a execução daquele serviço. Recomendamos ao usuário não modificar esses parâmetros de inicialização. A configuração dos fakes pode ser feita modificando seus artigos de configuração correspondentes, encontrados no subdiretório `conf`. São eles:

- `fakesquid.conf`: Arquivo de configuração do *fakesquid*, emulador do proxy Squid. O parâmetro de inicialização é `$bugsquid=" Squid/2.4 Stable3 ";`, que indica o *banner* da versão que deve aparecer nas respostas do fakesquid
- `fakesmtp.conf`: Arquivo de configuração do *fakesmtp*, emulador de servidores de correio. Possui os seguintes parâmetros: `$serveremul="sendmail";` indica qual servidor de correio será emulado. As opções válidas são: `exchange, sendmail, qmail` e `postfix`. Já `$logdir="logs/smtp";` indica o diretório de *log* (registro) do servidor de SMTP onde serão armazenados os arquivos de log e das mensagens enviadas com o endereço IP da máquina do agressor.
- `httpd.conf`: Arquivo de configuração do *fakehttpd*, emulador do Apache. Possui o parâmetro `$httpd="Apache/1.3.27";`, que indica a versão do Apache que deverá aparecer no banner nas respostas do fakehttp.
- `pop3.conf`: Arquivo de configuração do *fakepop3*, emulador de servidores POP3. Possui os seguintes parâmetros: `serveremul="qpopper";`, que indica qual servidor POP3 será emulado. As opções válidas são `teapop, qpopper` e `pop3`. Já `$logdir="logs/pop3.log";` indica o diretório em que ficarão os arquivos de log do serviço.
- `fakeftp.conf`: Arquivo de configuração do *fakeftp*, o emulador do servidor FTP *wuftp*. Os parâmetros de configuração são os seguintes: `$programaftp="wuftp";` indica o servidor FTP a ser emulado. Já `$conteudoftp="total 0\x0d\x0a";` indica para o fake qual conteúdo (chamado *honeytokens*) será exibido para o agressor.
- O fake *echo* não possui arquivo de configuração. O *fakepit* é utilizado para captura de assinaturas de worms, identificados por sua porta de entrada.

Listagem 4: rc.local

```
01 #!/bin/sh
02 #
03 # /etc/rc.d/rc.local:  Local system initialization script.
04 #
05 # Put any local setup commands in here:
06 #
07 #
08 echo “Iniciando o honeypot”
09 #
10 #Iniciando o snort
11 snort -c /etc/snort/snort.conf&
12 #
13 #Iniciando firewall
14 /etc/rc.d/./rc.firewall
15 #
16 # Mostrando regras de firewall
17 iptables -nL
18 #
19 echo “ok”
```

Configurando o iptables

Para finalizar o projeto, criaremos um pequeno script de iptables para nosso honeypot. Esse script irá liberar o acesso às portas dos serviços executados pelo honeypot. Veja a **listagem 3**.

Salve o arquivo com o nome de `rc.firewall` no diretório `/etc/rc.d/` e em seguida digite o comando `chmod 755 rc.firewall` para que possamos executá-lo a partir do script de inicialização `rc.local`.

Executando o Honeypot

Para executar o Honeyperl, simplesente execute o comando `perl honeyperl.pl`. Uma mensagem surge na tela informando que o programa está em execução. Para maior comodidade, rode o programa em segundo plano (*background*) com o comando `perl honeyperl.pl&`.

É possível adicionar parâmetros como `-h` (ajuda), `-v` (imprime no terminal avisos sobre os ataques mesmo que o arquivo de configuração determine o contrário) e `-l arquivo.log` (especifica um arquivo de log e sobrescreve o parâmetro equivalente no arquivo `honeyperl.conf`).

A partir desse instante o Honeypot escutará qualquer ataque que seja feito às portas ativas dos fakes em execução. Os logs ficarão armazenados no sub-diretório `logs`, em sub-diretórios identificados com o nome do serviço (`echo, ftp, httpd, squid`, etc). A estrutura do nome dos arquivos é:

```
logs/httpd/12-20-2004(18:15:23).log
```

Ou seja, *logs/serviço/mês-dia-ano(hora).log*. Dentro do arquivo temos informações como as mostradas a seguir, que ilustram um ataque ao fakehttpd:

```
Mon Dec 20 18:21:39 2004  fakehttpd log - Connection from 10.0.0.1:37378
get http 1.1 : Ataque WEB ! Tentativa de execucao de comando
```

Um toque final

Vamos deixar nosso sistema "pronto" para execução do honeypot logo após a inicialização. Edite o arquivo `/etc/rc.d/rc.local` para que se pareça com o mostrado na **listagem 4**. Agora, reinicie seu computador. Após o login, digite o comando: `perl /diretoriodohoneyperl/honeyperl.pl` e nosso honeypot estará pronto para o trabalho.

Considerações

Honeypots devem ser implementados de maneira cuidadosa, e é bastante interessante que seja feito um estudo de completo de viabilidade antes da instalação. Um Honeypot pode se tornar uma fonte enorme de informações, mas se mal-utilizado será uma brecha de segurança em qualquer sistema. ■

Sobre o autor

Antonio Marcelo é autor de 13 livros sobre Linux, mantenedor do projeto HoneypotBR, membro da IISFA e especialista em segurança com mais de 5 anos de experiência. Também é professor conferencista da cadeira de Segurança da Informação do curso de pós-graduação na Faculdade Estácio de Sá. Seu email de contato é amarcelo@plebe.com.br.

Informações

[1] SPITZNER, Lance. *Honeypots Tracking Hackers*. USA: Addison Wesley, 2003

[2] snort: www.snort.org

[3] Projeto HoneypotBR: www.honeypot.com.br

[4] Iptables: www.netfilter.org

[5] AMOROSO, Edward. *Intrusion Detection: An Introduction to Internet Surveillance, Correlation, Trace Back, Traps and Responses*. USA: Intrusion NetBooks, 1999.

[6] D.B. Moran, *Trapping and Tracking Hackers: Colletive Security for Survival in the Internet Age*, Third Information Survivability Workshop, IEEE Computer Society Press, October 2000.

[7] BAUMANN, R. Platner, C. - *Honeypots - Diploma thesis for masters in computer sience* Em inglês em www.rbaumann.net

[8] FONSECA, Marcelo, Antonio (2002/2003) *Anomalias de Pacotes - Partes I,II,III e IV*, revista Geek edições 26,27,28 e 29

[9] FYODOR, Nmap, 2001. Disponível em: www.insecure.org/nmap

[10] HONEYNET PROJECT TEAM. *Know Your Enemy*. USA: Addison Wesley, 2002.

[11] HANDLEY, Mark. *Network Intrusion Detection: Evasion Traffic Normalization, and End-to-End Protocols Semantics. Colletive Security for Survival in the Internet Age*, Third Information Survivability Workshop, IEEE Computer Society Press, October 2000.

[12] KLUG, David. *Honey Pots and Intrusion Detection*, SANS Institute, September 13 - 2001.

[13] NORTHCUTT, Stephen. *Network Intrusion Detection: An Analysis Handbook*. USA: New Riders Publishing, 1999

[14] STOLL. Cliff. *The Cuckoo's Egg: Tracking a Spy Through the Maze of Computer Espionage*, USA: Pocket Books, 2000

[15] STOLL. Cliff. *Stalking the Wiley Hacker*, Communications of the ACM, Vol 31 No 5, May 1988, pp. 484-497