Anais do 6º Fórum Internacional Software Livre - FISL
6º Workshop sobre Software Livre
Porto Alegre – RS – 2005
P. 245-250

Uma Infra-estrutura de Segurança Baseado em Logs para Plataformas UNIX/LINUX

Marcos Aurelio Pchek Laureano

FAE Centro Universitário
Rua 24 de Maio, 135 – 80230-080 – Curitiba – PR – Brasil

laureano@ppgia.pucpr.br

Resumo: *Os registros históricos ou logs são utilizados para diversos fins, normalmente como trilhas de auditoria. Com a utilização de algumas ferramentas livres é possível montar uma infra-estrutura de segurança baseada na verificação de logs, de forma a garantir que as políticas de segurança para a utilização de uma rede sejam cumpridas pelos funcionários. Este artigo apresenta uma proposta para servidores UNIX/Linux utilizando algumas ferramentas livres, em alguns casos, com poucas alterações.*

Palavras chaves: Segurança; Auditoria; Software Livre

Abstract: *The historical registers or logs are used for diverse objectives, normally as audit trail. With the use of some free tools it is possible to mount a security infrastructure based on the logs verification, of form to guarantee that the employees fulfill the politics of security for the use of a net. This article presents a proposal for UNIX/Linux servers using some free tools, in some cases, with few alterations.*

Keywords: Security; Audit; Free Software

1. Introdução

A tecnologia mudou as formas como se usam sistemas de informação. As possibilidades e oportunidades de utilização são muito mais amplas, assim como os riscos à privacidade e a integridade da informação. Portanto, é muito importante que mecanismos de segurança da informação sejam projetados de maneira a prevenir acessos não autorizados aos recursos e dados destes sistemas. Mesmo com a tecnologia disponível no momento, é praticamente impossível impedir que isso aconteça. Não existe um mecanismo único que forneça uma solução esse problema.

Esse trabalho descreve como uma infra-estrutura de segurança baseado em logs, foi implementada nas plataformas UNIX (AIX, HPUX, True64 e Solaris) e Linux utilizando software livre com algumas modificações.

2. Cenário Atual

A empresa HSBC Brasil S.A., com sede em Curitiba/PR, atuante no mercado financeiro, possui aproximadamente 250 servidores UNIX (distribuídos entre os sistemas AIX, HPUX, True64 e Solaris) e 1500 servidores Linux (Conectiva e Red Hat). A empresa tem equipes distintas para segurança e suporte destes servidores. Sempre que a equipe de suporte necessita realizar alterações de qualquer natureza, a

equipe responsável pela segurança é acionada para liberar o acesso ao usuário `root` (administrador) destas máquinas.

Para controlar e agilizar o suporte, inicialmente foi implantado o comando `sudo`[1] [Miller, 2004]; O comando `sudo` possibilita registrar em arquivos de log os comandos executados pelos seus usuários. Mas algumas tarefas exigiam que fosse utilizado o usuário `root` diretamente, utilizando-se o comando `su`[2] [Read Hat, 2003; GNU Project, 2001] para o acesso a este usuário.

Ambas as soluções continham falhas, pois era possível explorar vulnerabilidades dos comandos e assim executar tarefas não condizentes com o serviço previamente especificado, descaracterizando a liberação fornecida pela equipe de segurança. Como agravante, os poucos registros gerados eram armazenados localmente, sendo passíveis de alterações indevidas pelas pessoas que tinham o acesso como `root`.

3. Premissas do Projeto

Conforme [Wadlow, 2001], se o atacante tem acesso irrestrito ao sistema operacional, então qualquer informação proveniente deste sistema deve ser considerada suspeita. Assim, se a equipe de suporte tem acesso como `root`, as informações dos arquivos de logs não poderiam mais ser considerada confiável.

A solução seria armazenar todas as informações em outro equipamento onde somente a equipe de segurança tenha acesso, inclusive cabendo o suporte deste equipamento á equipe de segurança.

Outra premissa diz respeito a acompanhar e refazer todos os passos, se necessário, para fins de auditoria e restauração da integridade do sistema. Neste caso é necessário que todos os comandos fossem registrados, desde a entrada (`login`) até a saída (`logoff`) do sistema operacional.

4 Solução Adotada

O Projeto Honeynet [HoneyNet Project, 2001], utiliza o *shell* `bash` modificado para registrar todos os comandos digitados pelo usuário invasor. Naquele projeto, a versão do *shell* `bash` utilizada era a 2.02 e a função responsável por gravar os comandos no arquivo `.bash_history` foi modificada para executar a função `syslog`[3]. A mensagem gerada é classificada como Local e Informativa (`local5.info`, conforme padrão do `syslog`).

Na infra-estrutura adotada, as alterações foram portadas para a versão 2.05b do *shell* `bash` [GNU, 2003]. Ao ser portado, foram acrescidas outras informações para identificar o autor do comando (usuário efetivo e usuário real do sistema operacional) e o registro das informações é realizado apenas para os usuários com UID[4] menor que

[1] Este comando permite que o usuário execute outros comandos com poderes de administrador (`root`).
[2] Este comando permite acessar o perfil de outros usuários, sendo solicitado a senha do usuário que se deseja acessar.
[3] Esta função formata uma mensagem e envia para o *daemon* `syslogd`, o *daemon* padrão de registro de logs em sistemas Unix e Linux.
[4] UID – *User Identification*, número que identifica o usuário perante o sistema operacional.

100 (o UID do root é 0). A alteração ocorre na função add_history do *shell* bash, sendo acrescido ao início da função o bloco de código (Figura 1).

```
char s[20], s1[20];
memset(s, 0, sizeof(char)*20);
memset(s1, 0, sizeof(char)*20);
cuserid(s);
strncpy(s1, getlogin(), 20);

if (logme && getuid() < 100){
   if (strlen(string)<600) {
      syslog(LOG_LOCAL5 | LOG_INFO, "PWD=%s UID=%d LN=%s USR=%s
CM=%s", getenv("PWD"), getuid(), s1, s, string);
   }
   else {
      char trunc[600];
      strncpy(trunc,string,sizeof(trunc));
      trunc[sizeof(trunc)-1]='\0';
      syslog(LOG_LOCAL5 | LOG_INFO, "PWD=%s UID=%d LN=%s USR=%s
CM=%s(++TRUNC)",getenv("PWD"),getuid(),s1,s, trunc);
   }
}
```

Figura 1 – Trecho do código do *shell* bash modificado.

A função add_history já recebe na variável string o comando com os seus parâmetros que foi executado pelo bash.

O comando su foi alterado para solicitar sempre a senha do usuário, já que por padrão o usuário root tem acesso a qualquer outro usuário do sistema operacional sem precisar conhecer a senha deste usuário. Esta alteração impede um usuário com acesso de root execute operações indevidas utilizando a chave de outros usuários. Qualquer acesso ao root pelo comando su é registrado através da função syslog, pois o comportamento padrão do comando é registrar no arquivo sulog, e a equipe de segurança é notificada por e-mail.

Para garantir a integridade das logs, as mesmas são enviadas para um servidor com o sistema operacional ZLinux da SuSE [SUSE, 2004], instalado numa partição do IBM Mainframe com 20 GB de espaço. O arquivo /etc/syslog.conf, de todos os equipamentos, foi alterado para enviar todos os registros gerados pelos comandos bash e su para este servidor, através da inclusão da linha:

```
local5.info                @servidor
```

E no servidor de logs, o arquivo /etc/syslog.conf foi alterado para informar em qual arquivo serão registradas as informações:

```
local5.*                   /var/log/bash.log
```

No servidor de *logs*, foi implementado o syslog-ng [Scheidler, 2004] como *daemon* padrão de *logs*, principalmente pelo fato do mesmo permitir armazenar os registros em tabelas do MySql e ser mais confiável que o syslog padrão, entre outras funcionalidades. Para garantir a efetividade do processo, foi instalado o pacote OsHids [Os-Hids Project, 2004] para notificar, através de pesquisas por palavras-chaves, qualquer tentativa de comprometer a infra-estrutura, como por exemplo, alterações no

arquivo `/etc/syslog.conf` para não realizar mais o envio dos registros para o servidor ou tentativas de utilizar outro *shell* que não o `bash` (`csh` ou *C Shell*, por exemplo). O *daemon* do `syslog` é monitorado pelo `IBM Tivoli` [IBM, 2004], que garante a continua execução deste *daemon* e notifica qualquer comportamento anormal.

O resultado gerado pelo novo *shell,* e gravado pelo `syslog-ng,` é similar ao registro apresentado na figura 2.

```
Apr 13 18:49:58 guest bash: PWD=/root UID=0 LN=root USR=root CM=cd
/var/log/
Apr 13 18:49:59 guest bash: PWD=/var/log UID=0 LN=root USR=root CM=ll
Apr 13 19:06:51 guest -bashnew: PWD=/home/y0001 UID=0 LN=bob USR=root
CM=id

Apr 13 19:06:56 guest -bashnew: PWD=/home/y0001 UID=0 LN=bob USR=root
CM=ls -la
```

Figura 2 – Registro do *daemon* `syslog`.

Os registros da figura 2 demonstram como o *shell* `bash` modificado atua. O registro de `Apr 13 18:49:58` mostra que foi executado o comando `cd /var/log` pelo usuário `root`. O registro `Apr 13 19:06:51` mostra que o usuário `y0001` digitou o comando `id`, mas o *login name* (campo LN) é do usuário `bob`, o que demonstra que este usuário está utilizando o sistema com o perfil do usuário `y0001` (acessado através do comando `su`).

São realizadas cópias de segurança de todos os arquivos de *logs* (compactados) e banco de dados, estas cópias são armazenadas por 06 meses conforme política da empresa.

A figura 3 ilustra, de forma simples, a arquitetura de rede existente e como o servidor de *logs* ficou posicionado na estrutura. Na arquitetura existente, as verificações das *logs* do ambiente da DMZ possibilitam detectar possíveis invasões externas a empresa (internet). A empresa utiliza, internamente, rede *Ethernet* de 100 Megabits.

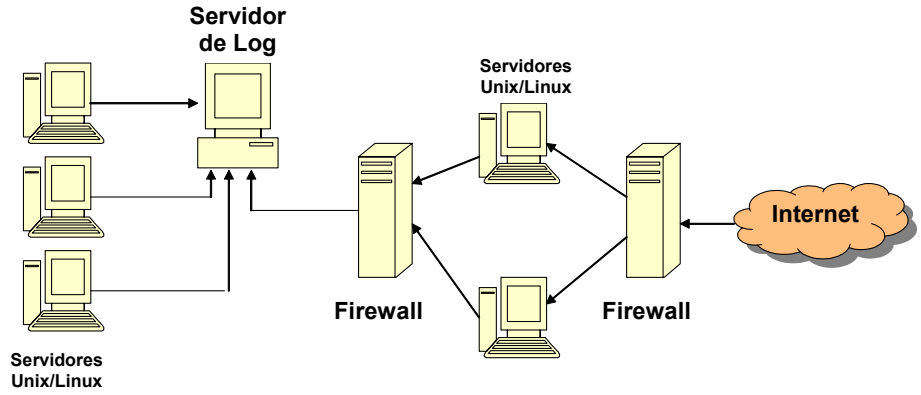

Figura 3 – Demonstração da Arquitetura de Rede

Anais do 6º Fórum Internacional Software Livre - FISL
6º Workshop sobre Software Livre
Porto Alegre – RS – 2005
P. 245-250

5 Resultados Obtidos

Após a implantação da infra-estrutura, foram detectadas várias ocorrências que caracterizavam o não cumprimento das políticas de segurança. As ocorrências são identificadas em até 05 minutos (da ocorrência até notificação por e-mail e acesso posterior ao servidor afetado), permitindo que a equipe de segurança consiga tomar as medidas cabíveis. Nestes casos foi realizada uma campanha de conscientização junto aos usuários, posteriormente será aplicada sanção disciplinar em caso de reincidência.

Os testes de performance e tráfegos de rede realizados indicaram que a solução é aceitável, tendo aumentando em apenas 1% o tráfego de rede e não sendo possível identificar perda de performance no sistema operacional.

Cabe ressaltar que o uso do usuário `root` não ocorre continuamente em todos os equipamentos simultaneamente, embora nos testes realizados foi considerado o uso simultâneo dos equipamentos para tarefas administrativas corriqueiras.

A principal limitação da proposta está relacionada com a monitoração dos comandos digitados. É necessário montar uma relação com os comandos que podem ser utilizados pelo usuário `root`. Para cada comando digitado, a relação deve ser consultada para verificação e notificação. Durante o tempo da detecção de uma ocorrência até notificação e verificação pela equipe de segurança (normalmente 3 a 5 minutos até que a equipe de segurança acesse o servidor afetado para acompanhar os procedimentos), pode ocorrer comandos que violem as políticas de segurança. Por exemplo: usuário `root` utilizando outro *shell*, como o ksh (*Korn Shell*), para evitar a detecção.

6. Conclusão

Este trabalho demonstrou como foi implementada uma infra-estrutura de segurança confiável através do uso de *logs*, utilizando as ferramentas padrões disponíveis nos sistemas operacionais com algumas alterações em seus códigos (benefício do software livre).

A infra-estrutura demonstrou sua viabilidade técnica e baixo custo operacional. Com a infra-estrutura implantada, várias violações a política de segurança da empresa foram detectadas e contidas pela equipe de segurança.

Os códigos fontes alterados e binários para as plataformas citadas no artigo podem ser encontradas em `www.ppgia.pucpr.br/~laureano`.

Referências

GNU Project (2001). “Shellutils – Command Line Utilities”. Disponível em: http://directory.fsf.org/shellutils.html. Acessado em 12/01/2005.

GNU Project. (2003). “GNU Bash – Table of Contents”. Disponível em: http://www.gnu.org/software/bash/manual/bash.html. Acessado em 10/01/2005.

HoneyNet Project. (2001) “Know Your Enemy: Revealing the Security Tools, Tactics and Motives of the Blackhat Community”. Addison-Wesley.

IBM (2004). “IBM Tivoli Software”. Disponível em: http://www-306.ibm.com/software/tivoli/. Acessado em: 15/12/2004.

Anais do 6º Fórum Internacional Software Livre - FISL
6º Workshop sobre Software Livre
Porto Alegre – RS – 2005
P. 245-250

Miller, T. (2004) “Sudo Main Page”. Disponível em http://www.courtesan.com/sudo/. Acessado em 15/01/2005.

Os-Hids Project (2004). “OsHids – Open source log analysis tool”. Disponível em: http://www.ossec.net/oshids/. Acessado em: 22/02/2005.

Red Hat (2003). “Red Hat Linux 9: Red Hat Linux Security Guide”. Disponível em: http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/security-guide/s1-wstation-privileges.html. Acessado em 15/01/2005.

Scheidler, B. (2004). “Syslog-ng reference manual”. Disponível em: http://www.balabit.com/products/syslog_ng/reference-1.6/syslog-ng.html/book1.html. Acessado em 22/02/2005.

SuSE (2004). “SUSE LINUX Retail Solution White Paper”. Disponível em: http://www.novell.com/collateral/4621408/4621408.pdf. Acessado em: 25/01/2005.

Wadlow, T. (2000) “Segurança de Redes – Projeto e Gerenciamento de Redes Seguras.” Campus, Rio de Janeiro – RJ.