Análise de Logs em Sistemas UNIX

CIRP - Centro de Informática de Ribeirão Preto - USP

MSc. Eng. Ali Faiez Taha
SCSUPOR - CIRP

Sumário

Capítulo 1

Análise de logs

1.1 Introdução

A monitoração do trabalho de servidores de rede sempre é acompanhada analisando-se arquivos e relatórios gerados por determinados programas.

Estes programas podem ser daemons de serviços ou programas dedicados. Estes relatórios são comumente denominados **arquivos de log** que vão mostrar a situação de cada serviço, mensagens que ajudarão a entender o que se passa com o servidor, mensagens de erros, alertas, etc.

Através destes relatórios, o Administrador de Sistemas vai executar determinadas tarefas de modo a manter o servidor em condições de funcionamento ou tomar decisões corretivas.

As máquinas **UNIX** utilizam um dos sistemas de **logging** mais simples, porém útil.

Os programas possuem duas opções principais quando se trata de gerar arquivos de **log**:

Arquivos de log gerados pelo processo Alguns programas tratam do seu próprio **logging**. Isso significa que seus arquivos de **log** contêm a saída apenas dessa origem. Os arquivos de **log** normalmente são determinados por meio de argumentos da linha de comandos ou arquivos de configuração, ou então são definidos dentro do próprio programa. Por exemplo, o **Apache WEB Server** possui um arquivo de log de acesso contendo os **URLs** atendidos (normalmente chamado de **acces_log**) e um arquivo de **log** de erro relacionando os problemas (páginas faltando, respostas de CGI inválidas, etc) que ele experimenta (normalmente chamado de **error_log**).

Mensagens syslog O modo mais comum de os programas registrarem informações é por meio do daemon **syslogd**. Esse é um programa cuja única finalidade é permitir um método comum de **logging** para programas divergentes. **Syslog** determina o que fazer com os **logs**, dependendo de duas coisas: a **facilidade syslog** e o **nível de logging**.

Todos os softwares tem sua própria maneira de gravar os arquivos de **log.**

1.2 O que são logs ?

Registro histórico das atividades de um dado programa/sistema.

Em geral, programas não interativos (servidores executando em **background**, **kernel** dos Sistemas Operacionais): Um dos benefícios do **UNIX** (e seus derivados) é fornecer mecanismos padronizados de **logging** (mensagens de execução de programas e processos) das atividades dos numerosos **daemons** e programas que estão sendo executados no sistema.

Estes **logs** podem ser usados para verificar erros (**debug**) do sistema, monitorar a sua utilização, cobrindo tudo, desde possíveis falhas de segurança até mensagens de alerta (**warning**) de possíveis problemas de **hardware**.

Às vezes é a única maneira de saber o que eles estão fazendo.

Logs gerados voluntariamente:

- A geração de **logs** normalmente pode ser desligada.
- Gerar **logs** úteis é característica de um bom programa servidor.

Logs são normalmente armazenados e apresentados em modo texto e

Permite fácil consulta usando qualquer editor de textos.

Se apresentam frequentemente com jargões e abreviaturas específicas que tornam a leitura incompreensível para um não-iniciado.

Às vezes, porém, requerem visualizadores específicos (exemplo: o **Event Viewer** do **Windows**).

1.3 O que é "logado" ?

É importante verificar a diferença entre os tipos de **logs** encontrados num **S.O. UNIX**

Basicamente há dois tipos de logs: **logs do sistema** e **logs dos aplicativos**.

Todos os Sistemas Operacionais apresentam os **logs** de sistema.

Nos **S.O. Windows** há aplicativos dedicados para a geração e interpretação dos logs do sistema, enquanto que nos **S.Os. Unixes** os arquivos de **log** (na sua maioria) são armazenados em formato de texto simples.

Os **logs** de aplicativos são dependentes das aplicações que estão sendo executadas e como estas aplicações estão configuradas para gerar os **logs**.

Nos **logs de sistema** serão encontradas mensagens e alertas (**warnings**) do **Kernel** que incluem informações dos módulos carregados, dados do **Sendmail**, que permite visualizar o caminho das mensagens que são processadas no sistema e mensagens sobre tentativas de logins efetuados com sucesso ou conexões que falharam.

Os **logs** de sistema são gerados pelo **Daemon "syslogd"**, que é carregado na inicialização do **UNIX**. O **syslogd** acessa mensagens em oito níveis de cada processo do sistema, tais como:

Kernel, sistema de E-Mail, programas de usuários configurados a usar o syslogd e programas de autenticação (o login).

Os níveis de mensagens são (em ordem crescente de prioridade):

debug
info
notice
warnig
err
crit
alert
emerg

Estes níveis são usados no arquivo **/etc/syslog.conf**, que faz com que o **syslogd** crie **logs** para diferentes tipos de informação.

O arquivo **/etc/syslog.conf** contém várias entradas, uma por linha, cada uma contendo dois campos separados por um ou mais espaços:

o nível de log e a **localização do arquivo de log.**

A lista dos níveis de **lo**g é formada por níveis de log separados por ponto-e-vírgula.

Estes níveis de logs são indicados por nomes, tais como **"mail", "kern" (para o kernel), "user" (para os usuários) e "auth" (para os programas de autenticação**).

Os pares **níveis de log** incluem:

mail.err : mensagens de erros geradas pelos servidores de **E-Mail**

***.info** : todas mensagens de informações

kern.emerg : mensagens de emergência do **Kernel**

1.4 Para que servem os logs ?

Finalidades principais:

Depuração de problemas (**debugging/troubleshooting**):

Informais, assume-se que sejam confiáveis e que não tenham sido adulterados**.**

Cômputo de estatísticas de uso e/ou performance:

Exemplos: **logs de Webservers, de Servidores de E-Mail, de FTP, etc.**

Fornecer trilhas de auditoria:

Tendem a ser mais formais, preferivelmente com detecção de adulteração, visando análise forense. Por exemplo: **Auditoria de um sistema comprometido.**

Billing/cobrança:

Serviços cobrados por volume/quantidade por período de tempo.

1.5 Sincronização dos relógios de rede

Muitos serviços de rede se utilizam deste recurso para sincronizar seus relógios.

O serviço **NTP (Network Time Protocol)** executa esta função.

Sistemas de **log** sincronizados com relógios padrões ajudam na verificação precisa dos **logs** de um servidor.

Recomenda-se a utilização de mais de um servidor **NTP**.

O configurador deste recurso é o comando ntpdate. A vantagem é a precisão (da ordem de segundos). Rápido e fácil de instalar, mas requer um servidor **NTP** que não saia do ar.

NTPd instalado localmente provê uma precisão de milissegundos.

Mais trabalhoso de instalar, mas pode ser tornado robusto com o uso de vários servidores.

1.6 Protocolo Syslog

É o protocolo e respectivo servidor do Unix clássico para receber mensagens de **log** via rede.

Suporta especificar os vários subsistemas (**facilities**) que originam a mensagem e um nível de prioridade.

Despacha a mensagem para um arquivo, para um determinado usuário logado na máquina, ou para outro servidor **syslog** em outra máquina.

Este protocolo usa a porta **514/UDP** e, geralmente faz parte da instalação default dos **S.O. UNIX**.

Se for desativado, ou não instalado, não haverá registro das mensagens de **log**.

O número de **logs** gerado é grande e pode ser configurado. Diversas configurações de **syslog** podem ser efetuadas. É bastante comum encontrar **clientes syslog**, **servidor syslog**, **servidores syslog em rede, etc.**

Numa rede estes elementos podem estar presentes, conectados através de **cabos seriais, crossover, switches ou Hubs**.

Também pode ser implementado um sistema de criptografia para o tráfego de **syslog**.

Utilizando o **syslog-ng (Next generation logging daemon)** é possível criar um **tunnel SSL** entre o **host** emissor e o receptor.

O **syslog** padrão pode ser substituído pelo **syslog-ng**, que será descrito no **Capítulo 3.**

1.7 Facilidades Syslog

Todas as mensagens **syslog** são marcadas com uma facilidade e nível específico. O arquivo **/etc/syslog.conf** permite especificar onde as mensagens entram. A facilidade **syslog** é simplesmente uma maneira de fazer com que um programa descreva em qual grupo de **logging** ele se encaixa.

As facilidades disponíveis são:

- **kern**: núcleo (**kernel**) do sistema operacional
- **user**: aplicação ou processo do usuários (**default**)
- **mail/news/UUCP**: subsistemas de correio eletrônico e notícias
- **cron**: executor de tarefas agendadas por horário
- **daemon**: servidores (**daemons**) do sistema
- **auth**: autorização, autenticação e controle de acesso
- **lpr**: subsistema de impressão
- **mark**: marcações de data/hora regulares
- **local0-local7**: para aplicações personalizadas
- **syslog**: mensagens internas geradas pelo próprio **syslog**
- **authpriv**: mensagens de autorização que não sejam do sistema em si
- **user**: mensagens genéricas no nível de usuário
- **"*"**: todas as acima exceto **mark**

1.8 Nível de logging de syslog

Os programas apanham cada entrada de **log** com um nível de **logging**, de modo que o daemon **syslog** possa informá-lo ou ignorá-lo, dependendo da configuração. Os níveis de mensagens são:

- **emerg**: situações de emergência/pânico
- **alert**: situações urgentes
- **crit**: situações críticas
- **warning**: advertências
- **notice**: situações incomuns que inspiram investigação

- **info**: informativos do estado normal do sistema
- **debug**: dados detalhados/prolixos para depuração
- **err:** situações de erros

1.9 Ações do Syslog

O registro das mensagens de **log** são estabelecidos no arquivo **/etc/syslog.conf**. O formato de cada linha é:

facilidade.nivel_do_log destino_do_log

Os campos são separados por tabulações.
Exemplo: daemon.notice /var/log/daemon.log
grava todos os **logs** para programas que estão usando a facilidade **daemon** e sejam de prioridade **notice** ou maior no arquivo **/var/log/daemon.log**. Pode-se especificar um asterisco (*****) para uma facilidade ou nível de **log** combinar com qualquer facilidade ou nível de **log**, respectivamente.

- **nomedearquivo**: acrescenta a mensagem para o arquivo especificado na máquina local
- **@nomeouip**: Repassa as mensagens para o servidor syslog na máquina especificada
- **user1,user2,&**: escreve a mensagem no console dos usuários especificados que estiverem conectados (se o usuário não estiver conectado, nada é escrito)
- *****: escreve a mensagem para todos os usuários conectados.

Exemplo de syslogd.conf:

Registra todas as mensgens do kernel no console

kern.* /dev/console

Registra tudo (exceto correio) de nível info ou maior

Não registra mensagens de autenticação privativas

***.info;mail.none;authpriv.none;cron.none /var/log/messages**

O arquivo authpriv tem acesso restrito

authpriv.* /var/log/secure

Registra todas as mensagens de correio em um só lugar

mail.* /var/log/maillog

Registre as atitudes do **cron**

cron.* /var/log/cron

```
# Todos recebem as mensagens de emergência; além disso, copie-as para outra máquina
*.emerg                                          *,@172.31.11.50

# Erros de correio e serviço de notícias em um arquivo especial
uucp,news.crit                                   /var/log/spooler

# Salve as mensagens de inicialização em um arquivo
local7.*                                         /var/log/boot.log

# Envia mensagens de nível crítico para o centalizador, mantendo

# uma cópia no arquivo de mensagens e avisando ao root
*.crit                                           /var/log/messages,@172.31.11.50,root
```

1.9.1 Armazenamento dos Dados

Os arquivos de log (nos **S.Os. UNIX**) são textos simples, crescem muito e a busca é sequencial .

Sem uma política de rotacionamento/descarte, os arquivos tendem a crescer sem limite.

O rotacionamento de **logs** será visto no **Capítulo 2.**

Os arquivos de **log** podem ser compactados e guardados e guardados em discos, **CD-ROMs, Fitas DAT, etc.**

Também podem ser utilizados Bancos de dados para se guardar os **logs**, mas estes costumam crescer mais rápido ainda.

Cuidados: Tamanho máximo fixo, conhecido desde o início: não cresce infinitamente

1.10 Arquitetura de coleta de Logs

Logging local em cada máquina Quando os **logs** são armazenados na própria máquina, sempre há o risco de usuários maliciosos que podem adulterar os **logs**.

Ferramentas para este fim são facilmente encontradas e são de fácil utilização.

Simples de utilizar e bastante perigosas. Adulteram os dados e comprometem a segurança.

Logging local + centralizado

- Se o **log** local for adulterado, ainda há a cópia no servidor de **log**.
- Espaço, tráfego, processamento, **backup** e resistência a ataques do servidor tornam-se questões mais complexas.

Para evitar estes tipos de problemas, aconselha-se que sejam utilizados **loghosts** bem dimensionados e bem protegidos.

Caracterísitcas de um **loghost**:

- servidor de **syslog**
- amplo espaço em disco
- política de rotacionamento

- **firewalling** próprio
- análise, sumarização e publicação

Desta forma, os arquivos de **log** podem ficar mais seguros, porém tenha sempre em mente que **nada é totalmente seguro**.

1.11 Centralizando logs

É bastante aconselhável a criação de um sistema de **loghost** que executa serviços muito limitados, apenas arquivando e processando dados de auditoria. A conexão com este sistema deve ser através de **SSH** ou outro protocolo com criptografia forte para acesso administrativo.

Configurações recomendadas:

- Dificulte o acesso à configuração do **syslog**
- **Colete logs via linhas seriais**
- Mantenha **logs, kernel e aplicações** em sistemas de arquivos distintos.
- Monitore o espaço em disco.
- Armazene dados em **CDs-worm**
- Documente os processos para gerência dos dados de auditoria.

Configure o sistema de logging no cliente para envio de syslog para o loghost.

Elabore uma política e configure S.Os., elementos ativos e aplicações para reportar os eventos nos quais esteja interessado.

1.12 O que deve ser "logado" ?

Dos diversos servidores que podem formar uma rede, todos devem ter um grau de importância. Classificação do que pode ser **logado**:

Servidores mais vulneráveis

- Web servers (públicos, intranet, extranet)
- Servidores de correio visíveis externamente.
- Estações dos admins, DBAs...

Dispositivos geralmente privilegiados

Qualquer sistema que abrigue dados corporativos

- Servidores de bancos de dados
- Repositório de código

1.13 Conteúdo do logging

Pode-se classificar os dados em:

- Eventos normais do dia-a-dia
- Assinaturas de ataques ou falhas dos sistemas
- Mensagens que voce não consegue identificar

Entre os eventos normais estão:

- Atividade autorizada
- Testes autorizados de segurança
- Erros ou problemas conhecidos
- Falsos positivos

Entre os eventos não críticos estão:

- Port scans
- Testes de vulnerabilidades não autorizados
- Tentativas mal sucedidas de comprometimento dos sistemas

Entre os eventos críticos estão:

- Tentativas bem sucedidas de comprometimento
- Ataques para redes de parceiros
- Queda de serviços devido a falhas de hardware ou de software
- Negação de serviço bem sucedida
- Mensagens desconhecidas

1.14 Falhas de Hardware

Muitas falhas de **Hardware** podem ser detectadas simplemente analisando os **logs** dos servidores.

No **UNIX**, geralmente em dispositivos **SCSI**, as mensagens encontradas podem indicar o mau funcionamento de um disco, de uma controladora **SCSI**, placas de rede, Drivers de **CD-ROM**, etc. Em sistemas **UNIX**, a verificação de muitos detalhes de **Hardware** é comum. Discos **IDE ATA IV** com cabos impróprios, falta de terminadores em cabos SCSI, placas de rede, placas de vídeo, memória, drivers de CD-ROM, etc, podem ser sinalizados nos arquivos de **log**.

Falhas de hardware frequentemente incluem palavras como:

- **error**
- **traceback**
- **panic**
- **dumping**
- **booting ...**
- além da popular: **file system full**

1.15 Logs de autenticação

Autenticação de serviços (FTP, TELNET, SSH, POP, etc)

- Sep 12 10:17:11 kuspy PAM_pwdb[17529]:authentication failure;(uid=0) -> tbird for ssh service
- Sep 12 10:17:12 kuspy sshd[17529]: log: Password authentication for tbird accepted.

Logs de autenticação frequentemente incluem palavras como:

- **authentication**
- **failure**
- **success**
- **login**
- **accepted**

Conexões permitidas/negadas em Roteador CISCO: Logs de autenticação frequentemente incluem palavras como:

- **Allowed**
- **Denied**
- **Access**
- **Refused**
- **TCP**
- **UDP**
- **ICMP** (e outros protocolos usados na rede)

1.16 Acesso à Web pages

- 172.20.1.54 - -[11/Feb/2000:20:39:10 -0800]"GET /img/cislogo.gif HTTP/1.0" 200 7607

Pode ser interessante buscar por:

Longas cadeias de dados sem sentido (pode indicar tentativa de buffer overflow)

Tentativas de executar scripts CGI não existentes

Caracteres especiais submetidos a formulários HTML

Tentativas de acesso a arquivos de:

- Senhas
- Configurações de **webserver**
- **ACLs** (access control files)

1.17 Coisas interessantes de procurar

WinNT/Win2K

- Usuário desconhecido ou senha rejeitada
- Reinício do sistema (**System Restart**)
- Descarte de eventos auditados
- Criação de contas
- Designação de direitos/permissões
- Novas relações/domínios de confiança

1.18 Removendo evidências

São ferramentas para adulterar arquivos de **log**. Largamente utilizadas por Hackers e Crackers.

As aplicações mais imediatas são remoção de vestígios de invasão, limpeza de pistas, violação dos arquivos de logs, modificação de conteúdo, etc.

Wipe

http://www.digitaloffense.net/worms/adore/lib_hack_hellno/wi pe-1.00/

Clássico do Unix, multiplataforma.

Seletivamente apaga vários logs (wtmp{,x}, utmp{,x}, lastlog, pacct, etc)

WinZapper, ClearLogs

http://ntsecurity.nu/toolbox/

Apagam seletivamente ou totalmente os logs do Windows

1.19 Analisadores para syslogs

logsurfer, logchecker Ferramentas que comparam certas expressões com textos encontrados nos arquivos de **log**.

As ferramentas para análise de **logs** serão descritas no **Capítulo 3**

- **http://www.cert.dfn.de/eng/logsurf/**

Win32 <-> Syslog

Backlog

- **http://www.intersectalliance.com/projects/BackLogNT/index.h tml**

NTsyslog

- **http://ntsyslog.sourceforge.net/**

Capítulo 2

Arquivos de log

Os arquivos de **log** (padrão **UNIX**) geralmente se encontram no diretórios **/var/log** ou **/var/adm**. Depende do Sistema Operacional que está sendo utilizado.

Os programas que geram **logs** podem ter seus arquivos de **logs** localizados nestes diretórios ou em outros locais, que devem ser especificados nos arquivos de configuração.

Há muitos softwares que, quando instalados a partir dos arquivos fontes, tem como argumento para compilação e instalação a localização dos arquivos de **log**.

No **UNIX**, os arquivos de **log** são especificados no **/etc/syslog.conf**, como será mostrado nese capítulo.

2.1 Descrição

Dependendo do **UNIX** utilizado, os arquivos trazem informações de muitos serviços.

Muitos serviços podem trazer seus próprios arquivos de **log**. Você pode criar seu próprio Software e utilizar o recurso de **logs**.

Dê uma olhada nos manuais relacionados a **syslog, syslogd, syslog.conf e logger .**

Para diferentes **UNIX**, as configurações são quase parecidas.

Muitos cuidados devem ser tomados em relação aos arquivos de **log**. Permissões e proprietários (usuário e grupo) devem ser bem especificadas.

Como o usuário **root** tem acesso a todos os arquivos do **UNIX**, não é uma boa prática fazer com que os **logs** sejam lidos apenas pelo **root.** Recomenda-se que seja criado usuários específicos para a análise dos logs de um **S.O. UNIX**.

O **root** só deve ser usado em casos de extrema necessidade.

As permissões devem ser bem seguras. Os softwares que geram arquivos de **log** fazem as recomendações mínimas necessárias. Não deixe de seguir estas regras. Todo cuidado é pouco.

Permissões fracas permitem que usuários indesejados alterem os arquivos de log e removam indícios e vestígios de ataques.

Exemplo de um arquivo de **log** de um sistema **UNIX** (**/var/log/messages**) mostrando informações num período de 28 minutos.

```
Jul 9 14:41:31 afthouse su: zemane to root on /dev/ttyp0
Jul 9 14:42:46 afthouse pppd[968]: pppd 2.3.5 started by zemane, uid 1001
Jul 9 14:42:47 afthouse pppd[968]: Connect: ppp0 <–> /dev/cuaa0
Jul 9 14:42:48 afthouse pppd[968]: local IP address 143.107.200.216
Jul 9 14:42:48 afthouse pppd[968]: remote IP address 143.107.200.1
Jul 9 14:43:23 afthouse su: zemane to root on /dev/ttyp1
Jul 9 14:43:36 afthouse su: zemane to root on /dev/ttyp1
```

```
Jul 9 14:44:38 afthouse su: zemane to root on /dev/ttyp1
Jul 9 14:46:02 afthouse kernel: sio0: 92 more interrupt-level buffer overflows (total 92)
Jul 9 14:46:42 afthouse kernel: sio0: 67 more interrupt-level buffer overflows (total 159)
Jul 9 14:59:20 afthouse pppd[968]: Connection terminated, connected for 16 minutes
Jul 9 15:04:37 afthouse su: zemane to root on /dev/ttyp2
Jul 9 15:09:03 afthouse pppd[1405]: pppd 2.3.5 started by zemane, uid 1001
Jul 9 15:09:03 afthouse pppd[1405]: Connect: ppp0 <-> /dev/cuaa0
Jul 9 15:09:04 afthouse pppd[1405]: local IP address 143.107.200.203
Jul 9 15:09:04 afthouse pppd[1405]: remote IP address 143.107.200.1
Jul 9 15:09:15 afthouse su: zemane to root on /dev/ttyp1
```

O trecho de texto acima mostra que o usuário **zemane** se conectou como root no terminal **/dev/ttyp0**, ou seja, o sistema informa que o usuário root está conectado (fez o **login**), uma mensagem sobre usuários conectados.

Em seguida o serviço **pppd** foi inicializado pelo usuário **zemane**.

O **kernel** envia mensagens sobre a utilização da **porta serial sio0** (**MODEM** na porta Serial). E assim por diante...

2.2 Arquivo de configuração syslogd.conf

Como descrito no capítulo anterior, arquivo **/etc/syslogd.conf** estabelece os arquivos e diretórios onde serão armazenados os arquivos de **log** do **UNIX**.

Para se ter uma idéia de como o **syslog** trabalha, eis um exemplo de um arquivo **/etc/syslog.conf** :

"arquivo.um"

```
# $FreeBSD: src/etc/syslog.conf,v 1.26 2003/04/23 13:08:31 des Exp $
# # Spaces ARE valid field separators in this file. However,
# other *nix-like systems still insist on using tabs as field
# separators. If you are sharing this file between systems, you
# may want to use only tabs as field separators here.
# Consult the syslog.conf(5) manpage.
*.err;kern.debug;auth.notice;mail.crit                          /dev/console
*.notice;authpriv.none;kern.debug;lpr.info;mail.crit;news.err   /var/log/messages
security.*                                                      /var/log/security
auth.info;authpriv.info                                         /var/log/auth.log
mail.info                                                       /var/log/maillog
lpr.info                                                        /var/log/lpd-errs
ftp.info                                                        /var/log/xferlog
cron.*                                                          /var/log/cron
*.=debug                                                        /var/log/debug.log
*.emerg                                                         *
# uncomment this to log all writes to /dev/console to /var/log/console.log
#console.info                                                   /var/log/console.log
# uncomment this to enable logging of all log messages to /var/log/all.log
# touch /var/log/all.log and chmod it to mode 600 before it will work
#*.*                                                            /var/log/all.log
# uncomment this to enable logging to a remote loghost named loghost
#*.* @loghost
# uncomment these if you're running inn
# news.crit                                                     /var/log/news/news.crit
# news.err                                                      /var/log/news/news.err
```

O exemplo acima é o arquivo (**/etc/syslog.conf**) de um Sistema Operacional **FreeBSD**.

Para os **S.Os. Linux**, o arquivo (**/etc/syslog.conf**) é bem parecido.

Eis um exemplo:

"arquivo.dois"

```
# /etc/syslog.conf Configuration file for syslogd.
# For more information see syslog.conf(5) manpage.
# First some standard logfiles. Log by facility.
#
auth,authpriv.*                                          /var/log/auth.log
*.*;auth,authpriv.none                                     /var/log/syslog
#cron.*                                                   /var/log/cron.log
daemon.*                                                /var/log/daemon.log
kern.*                                                   /var/log/kern.log
lpr.*                                                     /var/log/lpr.log
mail.*                                                /var/log/mail/mail.log
user.*                                                   /var/log/user.log
uucp.*                                                   /var/log/uucp.log
#
# Logging for the mail system. Split it up so that
# it is easy to write scripts to parse these files.
#
mail.info                                          /var/log/mail/mail.info
mail.warn                                          /var/log/mail/mail.warn
mail.err                                            /var/log/mail/mail.err
# Logging for INN news system
# news.crit                                        /var/log/news/news.crit
news.err                                            /var/log/news/news.err
news.notice                                      /var/log/news/news.notice
#
# Some 'catch-all' logfiles.
#
*.=debug;\
   auth,authpriv.none;\
   news.none;mail.none -/var/log/debug
*.=info;*.=notice;*.=warn;\
   auth,authpriv.none;\
   cron,daemon.none;\
   mail,news.none -/var/log/messages
#
# Emergencies are sent to everybody logged in.
#
*.emerg *
#
# I like to have messages displayed on the console, but only on a virtual # console I usually leave idle.
#
#daemon,mail.*;\
# news.=crit;news.=err;news.=notice;\
# *.=debug;*.=info;\
# *.=notice;*.=warn /dev/tty8
# The named pipe /dev/xconsole is for the 'xconsole' utility. To use it,
# you must invoke 'xconsole' with the '-file' option:
#
# $ xconsole -file /dev/xconsole [...]
#
# NOTE: adjust the list below, or you'll go crazy if you have a reasonably busy site..
#
daemon.*;mail.*;\
   news.crit;news.err;news.notice;\
   *.=debug;*.=info;\
   *.=notice;*.=warn
```

```
    |/dev/xconsole
```

Um trecho importante a se observar:

```
*.=info;*.=notice;*.=warn;\
  auth,authpriv.none;\
  cron,daemon.none;\
  mail,news.none -/var/log/messages
```

Estas linha mostram que os **logs** relativos a **info, notice e warn** dos serviços de autenticação **(auth), cron e mai**l serão colocados no arquivo **/var/log/messages**.

Não serão colocados no arquivo **/var/log/messages** os **logs** relativos a **"info", "notice" e "warn"** dos serviços **authpriv, daemon e news**.

Este trecho é indicado por : **authpriv.none, daemon.none e news.none**

Observando o **arquivo.dois**, todas as mensagens de **log** referentes a **"mail"** serão colocadas no arquivo **/var/log/maillog**

mail.* -/var/log/mail/mail.log

Há também outras mensagens de **logs** relacionadas ao serviço de **E-Mail:**

```
#
# Logging for the mail system. Split it up so that
# it is easy to write scripts to parse these files.
#
mail.info                                   /var/log/mail/mail.info
mail.warn                                   /var/log/mail/mail.warn
mail.err                                    /var/log/mail/mail.err
```

O arquivo de configuração de **logs** (**/etc/syslog.conf**) encontrado no **FreeBSD** (**arquivo.um**) é menor que o arquivo (**/etc/syslog.conf**) encontrado no **Linux**.

Faça uma comparação entre eles e veja as semelhanças, diferenças, etc.

Como mostrado e detalhado anteriormente, as mensagens de **log** são separadas em diferentes arquivos.

O propósito é manter maleável o tamanho de cada arquivo de **log** e facilitar a busca de mensagens nos arquivos, rastrear os arquivos e procurar por determinados padrões de textos tais como: **"permission denied"**, **"syntax error on line 123"**, **"user unkown"**, etc.

Se todas as mensagens de **log**, dos diferentes níveis, fossem colocadas em apenas em um arquivo, com certeza a dificuldade de rastrear e examinar este grande arquivo seria comprometida.

A maneira de armazenar as mensagens de **log** pode variar entre os diversas distribuições **Linux, FreeBSD** e outros **Unixes**.

As modificações no arquivo **/etc/syslog.conf** podem ser efetuadas somente pelo administrador do sistema (**usuário root**).

Após as modificações o **daemon syslog** deverá ser reinicializado.

kill -HUP 'cat /var/run/syslogd.pid'

2.3 Alguns arquivos importantes

- /var/log/syslog
- /var/log/messages
- /var/log/sulog

- /var/log/xferlog
- /var/log/wtmp, /var/run/utmp
- /var/log/auth.log
- outros (no diretório /var/log)

Interpretar alguns dos arquivos acima consiste em apenas visualizar seus conteúdos, pois são arquivos texto simples. Alguns são do tipo **binário** e exigem comandos especiais para fazer a sua interpretação.

Com os comandos **cat, more ou less** o conteúdo dos arquivos texto pode facilmente ser examinados. Com os comandos **zcat, zmore ou zless**, os arquivos **zipados (*.zip, *.gz)** podem ser visualizados.

O mesmo pode ser feito com editores de textos tais como o **vi, pico, nano, ae, vin, ex, etc.**

Deve-se tomar cuidado com o final de cada linha do arquivo texto. É fácil confundir e não perceber os finais de linha e considerar algumas linhas como sendo uma única linha. Para evitar isso é sempre recomendável que se use os editores de textos. Pode-se usar também o comando **view**.

O **editor view** não permite alteração do arquivo texto. É bom para editar e não modificar o conteúdo do arquivo.

Para acompanhar o crescimento de um arquivo de **log** (texto simples) use os comandos **head** e **tail**.

O **head** mostra as linhas iniciais de um arquivo. O **tail** mostra as últimas linhas de um arquivo texto.

O **tail -f arquivo** permite que se veja o contante crescimento de um arquivo texto simples.

Para interpretar arquivos de **log** que estão no formato **binário**, deve-se usar ferramentas apropriadas.

Como exemplo, vamos fazer uma breve interpretação do comando **last**, que mostra os usuários conectados, os que se conectaram ao servidor, datas, duração da conexão, etc.

Este comando ordena as entradas no arquivo, relacionando os tempos de login e logout. Se invocado sem argumentos, o comando **last** exibe toda a informação contida no arquivo.

O arquivo **wtmp** deve ser examinado manualmente através do comando **last**. Este arquivo **não é um texto puro**, é um arquivo tipo **data**, ou seja, **binário.**

O arquivo **/var/log/utmp** (veja **man utmp**) também é um arquivo **binário**.

O comando **last** é usado para identifiar os usuários que já se conectaram ou estão conectados.

O arquivo **wtmp** mostra os usuários que se conectaram. O arquivo **utmp** mostra os usuários que estão conectados.

Pode-se usar o comando **last** com a opção **-f** e especificando o **wtmp** ou o **utmp** como parâmetro.

Exemplos :

$> last -f /var/run/utmp —> vai mostrar o usuários conectados

$> last -f /var/log/wtmp —> vai mostrar os usuários conectados e os que se conectaram.

Para maiores detalhes **: man last (ou info last)**

O **syslog** é um mecanismo que permite que qualquer comando registre mensagens de erro e informativas na console do sistema e/ou em um arquivo.

Normalmente mensagens de erro são gravadas no arquivo **/var/log/messages** juntamente com a data e hora em que foram gravadas. Os arquivos de **log** encontrados no diretório **/var/log** possuem permissões de leitura e escrita somente para o **root** e permissão de leitura para os demais usuários. Muitos podem ser consultados apenas pelo **root**.

Muitos **UNIX** apresentam subdiretórios dentro do **/var/log**. Alguns subdiretórios guardam os **logs** para serviços bem específicos, tais como o Servidores de E-Mail (Sendmail, Postfix, QMail, etc).

Cada distribuição **UNIX** apresenta suas particularidades. A localização dos arquivos de **log** de cada programa depende das configurações destes.

Arquivos padrões de **logs** são armazenados no diretório **/var/log**.

Pode-se modificar os nomes dos arquivos e sua localização. O trecho abaixo faz o servidor de **DNS** armazenar os **logs** no arquivo **/var/log/quartzo.named.log.**

Os níveis de **log** são mais detalhados.

```
logging {
channel my_default {
file "/var/log/quartzo.named.log";
severity info;
print-time yes;
print-category yes;
print-severity yes; };
category default { my_default; default_debug; };
category panic { my_default; default_stderr; };
category packet { default_debug; };
category eventlib { default_debug; };
category lame-servers { null; }; }
```

2.4 Rotacionando os logs

O objetivo é fazer com que os arquivos de **log** sejam preservados, fazer um **backup** e criar novos arquivos frequentemente.

Com o rotacionamento os arquivos de **log** são **zipados**, renomeados e novos arquivos são criados.

As informações mais velhas ficam nos arquivos compactados.

Se não se faz o rotacionamento, o tamanho do arquivo tende a crescer muito e a sua manipulação torna-se praticamente impossível.

A maneira mais simples é remover os arquivos de **log** e reinicializar o **daemon syslogd**.

```
$> rm /var/log/messages
$> kill -HUP `cat /var/run/syslogd.pid`
```

Este método é bem aplicado num servidor **Linux** que não esteja em rede, ou num simples **Desktop com Linux**.

Em servidores Unix os arquivos de **log** precisam ser preservados para uma análise futura, uma auditoria, rastreamento de brechas de segurança, etc.

Neste caso a estratégia é diferente. **Os arquivos de log deverão ser mantidos.**

Deve-se renomear os arquivos antigos e criar arquivos novos.

Os arquivos antigos podem ser compactados e guardados em sistemas de **backup**.

```
$> mv /var/log/messages /var/log/messages.1
$> kill -HUP `cat /var/run/syslogd.pid`
```

Se se pretende criar duas gerações de histórico de **logs**, será necessário mover a primeira geração de arquivos para a segunda geração e, então, mover a geração atual para a primeira geração.

```
$> mv /var/log/messages.1 /var/log/messages.2
$> mv /var/log/messages /var/log/messages.1
$> kill -HUP `cat /var/run/syslogd.pid`
```

Aconselha-se a empacotar os arquivos de **log** antigos:

$> gzip /var/log/messages.1
$> gzip /var/log/messages.2

Em muitos **UNIX** este procedimento é automatizado.
No **Linux Debian**, e em outras distribuições, os scripts estão nos diretórios **/etc/cron.daily, /etc/cron.weekly** e **/etc/cron.monthly**.
Estes **scripts** devem ter permissão de execução.
Verifique estes arquivos e examine os **shell scripts** que realizam o rotacionamento dos **logs**.

2.5 Exemplos de arquivos de log

A seguir são mostrados alguns exemplos de arqivos de **log:**

- **$> cat /var/log/vsftpd.log | grep Debian**

Arquivo de log do servidor VSFTP, instalado num servidor **Linux Debian**

Mon Mar 15 12:32:22 2004 6 143.107.200.102 317706 /pub1/Cursos.CIRP/Curso.Adm.Linux.Debian.CIRP.pdf b _ o a mozilla@example.com ftp 0 * c

Mon Mar 15 18:05:33 2004 4 143.107.200.102 281448 /pub1/Cursos.CIRP/Curso.Adm.Linux.Debian.CIRP.quase.final.22.Julho.2003.pdf b _ o a mozilla@example.com ftp 0 * c

Mon Mar 15 18:11:47 2004 5 143.107.200.102 281448 /pub1/Cursos.CIRP/Curso.Adm.Linux.Debian.CIRP.quase.final.22.Julho.2003.pdf b _ o a -wget@ ftp 0 * c

Mon Mar 15 18:17:11 2004 5 143.107.200.102 317706 /pub1/Cursos.CIRP/Curso.Adm.Linux.Debian.CIRP.pdf b _ o a mozilla@example.com ftp 0 * c

- **$> cat /var/log/vsftpd.log.* | grep BSD**

Fri Mar 12 21:09:29 2004 1 200.187.167.146 9037 /FreeBSD/Dicas/experiencia.com.fortran b _ o a mozilla@example.com ftp 0 * c

Mon Mar 1 17:44:46 2004 1 200.222.74.74 9037 /FreeBSD/Dicas/experiencia.com.fortran b _ o a Squid@ ftp 0 * c

Tue Mar 2 11:46:52 2004 1 200.225.73.100 173 /pub1/FreeBSD/releases/i386/supfile b _ o a libramar.png@libramar.com.br ftp 0 * c

Thu Mar 4 15:05:53 2004 53 217.148.68.113 1480512 /pub1/OpenBSD3.2/OpenBSD32-i386-base.iso b _ o a Squid@ ftp 0 * i

Thu Mar 4 15:45:13 2004 3 200.181.17.112 451 /pub1/FreeBSD/ISO-IMAGES-i386/4.7/pegar b _ o a mozilla@example.com ftp 0 * c

Thu Mar 4 15:45:30 2004 1 200.181.17.112 451 /pub1/FreeBSD/ISO-IMAGES-i386/4.7/pegar b _ o a mozilla@example.com ftp 0 * c

Thu Feb 26 17:29:33 2004 1 143.107.70.187 183 /pub1/OpenBSD3.2/pegar b _ o a mozilla@example.com ftp 0 * c

Thu Feb 26 17:29:46 2004 1 143.107.70.187 75 /pub1/OpenBSD3.2/pegar2 b _ o a mozilla@example.com ftp 0 * c

Thu Feb 26 17:29:52 2004 1 143.107.70.187 65 /pub1/OpenBSD3.2/MD5SUM b _ o a mozilla@example.com ftp 0 * c

Fri Feb 20 10:31:11 2004 31 161.24.64.254 1136012 /pub1/OpenBSD3.2/OpenBSD32-i386-base.iso b _ o a IE40user@ ftp 0 * i

Fri Feb 20 10:31:34 2004 22 161.24.64.254 1004940 /pub1/OpenBSD3.2/OpenBSD32-i386-base.iso b _ o a IE40user@ ftp 0 * i

Fri Feb 20 10:32:09 2004 34 161.24.64.254 1519172 /pub1/OpenBSD3.2/OpenBSD32-i386-base.iso b _ o a IE40user@ ftp 0 * i

Fri Feb 20 13:29:49 2004 14698 200.161.147.235 645169152 /FreeBSD/5.1-RELEASE/5.1-RELEASE-i386-disc1.iso b _ o a anon@ ftp 0 * c

- **$> cat /var/log/vsftpd.log.* | grep Slackware**

Sat Mar 13 23:20:22 2004 1 201.4.162.96 184 /Slackware9.0/slackware/l/libxml2-2.5.4-i386-1.txt b _ o a mozilla@example.com ftp 0 * c

Sat Mar 13 23:28:46 2004 491 201.4.162.96 1434976 /Slackware9.0/slackware/l/libxml2-2.5.4-i386-1.tgz b _ o a mozilla@example.com ftp 0 * c

Sat Mar 13 23:50:56 2004 156 201.4.162.96 629704 /Slackware9.0/slackware/l/libtiff-3.5.7-i386-3.tgz b _ o a mozilla@example.com ftp 0 * c

- **$> cat /var/log/vsftpd.log.* | grep Mandrake**

Sun Mar 7 05:53:05 2004 1 200.171.10.214 7912 /Mandrake9.1/INSTALL.txt a _ o a anon@ ftp 0 * c

Sun Feb 22 10:28:36 2004 2 200.158.225.44 143310 /Mandrake9.1/pkg-9.1-Bamboo-i586.idx b _ o a Squid@ ftp 0 * c

Wed Feb 18 15:51:30 2004 201 200.201.164.11 9698952 /Mandrake9.1/ISO/Mandrake91-cd1-inst.i586.iso b _ o a proxyuser@proxy.caixa ftp 0 * i

- **$> tail -5 /var/log/dns.log**

20-Mar-2004 17:38:26.414 maintenance: info: Cleaned cache of 2163 RRsets

20-Mar-2004 17:38:26.483 statistics: info: USAGE 1079804306 1079534305 CPU=31.9493u/21.5371s CHILDCPU=0u/0s 20-Mar-2004 17:38:26.483 statistics: info: NSTATS 1079804306 1079534305 TYPE0=431 A=125083 NS=67 CNAME=8 SOA=1870 PTR=167113 MX=19477 AAAA=23608 SRV=1981 A6=2201 ANY=27473

20-Mar-2004 17:38:26.483 statistics: info: XSTATS 1079804306 1079534305 RR=213515 RNXD=74194 RFwdR=59395 RDupR=593 RFail=4017 RFErr=8849 RErr=1851 RAXFR=0 RLame=7432 ROpts=0 SSysQ=89629 SAns=398279 SFwdQ=82779 SDupQ=127745 SErr=5 RQ=370758 RIQ=0 RFwdQ=82779 RDupQ=12856 RTCP=1548 SFwdR=59395 SFail=265 SFErr=0 SNaAns=122215 SNXD=170870 RUQ=0 RURQ=0 RUXFR=0 RUUpd=484

20-Mar-2004 17:54:06.805 default: info: Response from unexpected source ([149.174.211.3].9052) for query "dns-02.ns.aol.com IN AAAA" 20-Mar-2004 17:56:40.521 update-security: notice: denied update from [143.107.200.190].2365 for "cirp.usp.br" IN

20-Mar-2004 17:56:40.550 update-security: notice: denied update from [143.107.200.190].2370 for "200.107.143.in-addr.arpa" IN 20-Mar-2004 17:57:01.205 default: info: ns_forw: query(3.186.53.195.in-addr.arpa) No possible A RRs

- **$> zcat /var/log/maillog.7.gz | grep virus**

Mar 12 21:50:13 quartzo sendmail[90697]: i2CLoDnB090697: Authentication-Warning: quartzo.cirp.usp.br: vscan set sender to virusalert@cirp.usp.br using -f

Mar 12 21:50:13 quartzo sendmail[90697]: i2CLoDnB090697: from=virusalert@cirp.usp.br, size=1269, class=0, nrcpts=1, msgid=<VAjPxmbCYF@quartzo.cirp.usp.br>, relay=vscan@localhost

Mar 12 21:50:13 quartzo sendmail[90697]: i2CLoDnB090697: to=virusalert@cirp.usp.br, delay=00:00:00, mailer=esmtp, pri=31269, dsn=4.4.3, stat=queued Mar 12 21:50:13 quartzo sendmail[90698]: i2CLoDS5090698: Authentication-Warning: quartzo.cirp.usp.br: vscan set sender to virusalert@cirp.usp.br using -f

Mar 12 21:50:13 quartzo sendmail[90698]: i2CLoDS5090698: from=virusalert@cirp.usp.br, size=666, class=0, nrcpts=1, msgid=<VRjPxmbCYF@quartzo.cirp.usp.br>, relay=vscan@localhost

Mar 12 21:50:13 quartzo amavis[90640]: (jPxmbCYF) INFECTED (Worm.SomeFool.Gen-2), <alexmgarcia2002@hotmail.com> -> <usuario@cirp.usp.br>, quarantine virus-20040312-215013-jPxmbCYF, Message-ID: , Hits:

Estas linhas mostram os e-mails recebidos com vírus. O usuário **virusalert@cirp.usp.br** foi notificado.

- **$> zcat /var/log/messages.8.gz**

Ese trecho mostra problemas de **Hardware**. Os **S.O. UNIX** fazem identificação detalhada dos problemas de **Hardware** e reportam em arquivos de **log.** O trecho inicial mostra o problema de **reset** constante na controladora **SCSI** e o trecho final não mostra estes problemas.

Aug 13 19:02:18 linorg kernel: SCSI bus is being reset for host 0 channel 0.

Aug 13 19:02:18 linorg kernel: (scsi0:0:0:0) Synchronous at 20.0 Mbyte/sec, offset 15.

Aug 13 19:02:19 linorg kernel: (scsi0:0:1:0) Synchronous at 20.0 Mbyte/sec, offset 15.

Aug 13 19:02:20 linorg kernel: scsi0 channel 0 : resetting for second half of retries.

Aug 13 19:02:20 linorg kernel: SCSI bus is being reset for host 0 channel 0.

Aug 13 19:02:20 linorg kernel: (scsi0:0:1:0) Synchronous at 20.0 Mbyte/sec, offset 15.

Aug 13 19:02:23 linorg kernel: scsi0 channel 0 : resetting for second half of retries.

Aug 13 19:02:23 linorg kernel: SCSI bus is being reset for host 0 channel 0.

Aug 13 19:02:23 linorg kernel: (scsi0:0:1:0) Synchronous at 20.0 Mbyte/sec, offset 15.

Aug 13 19:02:47 linorg kernel: (scsi0:0:0:0) Synchronous at 20.0 Mbyte/sec, offset 15.

Aug 13 19:13:25 linorg – MARK –

Aug 13 19:33:25 linorg – MARK –

Aug 13 19:53:25 linorg – MARK –

Aug 13 20:13:25 linorg – MARK –

Aug 13 20:33:25 linorg – MARK –

- **$> cat /var/log/apache/access.log | grep "cmd.exe"**

Trechos do arquivo de access.log do Apache Web Server.

61.115.118.197 - - [14/Mar/2004:07:47:35 -0300] "GET /scripts/..%c1%1c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 302 229

61.115.118.197 - - [14/Mar/2004:07:47:35 -0300] "GET /NULL.printer HTTP/1.0" 302 229

A primeira linha mosta atentativa de se executar o comando **dir**, do DOS.
A segunda linha mostra atentativa de se enviar algo para a impressora.

- **$> cat /var/log/apache/error.log | tail -5**

[Sun Oct 5 06:25:19 2003] [error] [client 64.68.80.44] File does not exist:
/raid/Conectiva7/Disk2/doc/howto/Online-Troubleshooting-HOWTO

[Sun Oct 5 06:25:39 2003] [error] [client 66.77.73.72] File does not exist:
/raid/Conectiva7/Disk2/doc/guias/guia_de_instalacao/node215.html

[Sun Oct 5 06:25:52 2003] [error] [client 66.77.73.72] File does not exist: /raid/Slackware8.0//bootdsks.144/optics.i

[Sun Oct 5 06:26:43 2003] [error] [client 64.68.80.41] File does not exist:
/raid/Conectiva7/Disk1//conectiva/cncimage/etc/X11/xkb/symbols/ua

[Sun Oct 5 06:27:20 2003] [notice] SIGUSR1 received. Doing graceful restart

- **$> tail -9 /var/log/auth.log**

Mar 22 10:15:52 quartzo sshd[47934]: Accepted keyboard-interactive/pam for rubens from 143.107.200.46 port 1315 ssh2
Mar 22 11:09:46 quartzo sshd[48811]: Accepted keyboard-interactive/pam for aftaha from 143.107.207.36 port 1029 ssh2
Mar 22 11:14:57 quartzo su: aftaha to toor on /dev/ttyp0
Mar 22 11:45:09 quartzo sshd[49502]: Accepted keyboard-interactive/pam for aftaha from 143.107.207.36 port 1031 ssh2
Mar 22 11:47:56 quartzo login: login on ttyv0 as rubens Mar 22 14:30:49 quartzo sshd[51096]: Accepted keyboard-interactive/pam for aftaha from 143.107.200.102 port 1171 ssh2
Mar 22 14:30:54 quartzo su: aftaha to toor on /dev/ttyp0
Mar 22 15:29:45 quartzo sshd[87]: Server listening on 0.0.0.0 port 22.
Mar 22 15:29:54 quartzo webmin[227]: Webmin starting
Mar 22 15:45:08 quartzo sshd[87]: Server listening on 0.0.0.0 port 22.

Capítulo 3

Ferramentas para análise de logs

As mais comuns são:

- **LogSentry** (Analisa expressões regulares nos arquivos de log do UNIX)
- **Logcheck** (sumariza os logs do UNIX e envia resultados por E-Mail)
- **Logmuncher** (variação do Logcheck)
- **Logsurf** (logs do UNIX, capaz de criar regras dinâmicas)
- **Swatch** (Simple Watch Dog, para arquivos de log do UNIX)
- **Sec** (correlacionador de eventos simples)
- **Lire** (logs do UNIX, de servidores WEB, etc)
- **Webalizer** (logs de Servidores WEB)
- **WebLog** (logs de Servidores WEB e Proxy)
- **Analog** (logs de Servidores WEB)
- **Calamaris** (logs do Servidor Proxy Squid)
- **Logguard (**sumariza os logs do UNIX e envia resultados por E-Mail)

Mais ferramentas podem ser encontadas nos seguintes **URLs**:
http://fmg-www.cs.ucla.edu/fmg-members/geoff/logmuncher.html,
http://www.hotscripts.com/Tools_and_Utilities/Log_Analyzers/

Uma das tarefas do administrador de sistemas é a monitoração da segurança.

Esta tarefa envolve o exame de arquivos de **log** para detectar acessos não autorizados, bem como a monitoração de falhas de segurança.

As contas devem ser monitoradas periodicamente de modo a verificar dois eventos: usuários que se conectam quando não devem (por exemplo, tarde da noite ou quando estão de férias) e usuários executando comandos que normalmente não deveriam usar.

O arquivo **/var/log/lastlog** (no **Linux**) registra o **login** mais recente de cada usuário do sistema.

A mensagem impressa no terminal a cada vez que um usuário se conecta utiliza a data armazenada no arquivo **lastlog**

Last login: Sat Mar 10 10:50:48 from **hostname.servidor.br**.

A data do último **login** relatada pelo comando **finger** (em desuso por questões relacionadas a vulnerabilidades) também usa estes dados.

Os usuários devem ser alertados a inspecionar esta data para certificarem-se de que não foi efetuado nenhum acesso não autorizado às suas contas e, caso positivo, alertar o Administrador de Sistemas para o ocorrido.

Algumas ferramentas disponíveis facilitam a interpretação dos arquivos de **log**.

Estas ferramentas possuem em seus arquivos de configuração instruções para procurar certas palavras, **strings** ou textos nos arquivos de **log** e enviar algum tipo de sinal para o Administrador de Sistemas. Por exemplo, se encontrar expressões como **access denied** em certos arquivos de **log**, um **e-mail** deverá ser enviado ao Administrador. Ele vai receber este **e-mail** e tomar alguma providência.

Basicamente esta é a função das ferramentas para **Análise de logs**.

A seguir serão descritas algumas ferramentas disponíveis para os **S.O. UNIX.**

3.1 LogSentry

É um verificador de **log** no estilo **cron**. Usa vários arquivos contendo expressões regulares **egrep** simples e os combina com as linhas no arquivo de **log** para determinar se um relatório deve ser feito. Os relatórios são remetidos ao **root** ou a outro usuário. Vários arquivos contêm as expressões regulares usada pelo **LogSentry**, como mostrado abaixo:

logcheck.hacking:

Expressões que definitivamente indicam atividade de invasão. Quaisquer mensagens que combinam são remetidas com um cabeçalho antipático, para chamar a atenção imediatamente.

logcheck.violations:

Expressões que indicam atividades impróprias, mas não tão sérias como aquelas em **logcheck.hacking**

logcheck.violations.ignore:

Expressões que são realmente benignas**.** Se uma linha combinar com uma regra em **logcheck.violations**, mas também combinar com uma regra em **logcheck.violations.ignore**, ela não será informada. Por exemplo, esse arquivo lhe permite apanhar mensagens contendo **refused** (como **TCP connection refused)** sem informar mensagens inocentes, como a possibilidade de Sendmail se conectar a um servidor de correio (que cria uma mensagem com **stat=refused**). Também usado para eliminar falsos positivos.

logcheck.ignore:

Se nenhuma combinação tiver sido feita até aqui, a linha será informada, a menos que haja uma combinação no arquivo **logcheck.ignore.**

LogSentry vem com padrões **default** embutidos de **logs** dos ataques de **Internet Security Scanner (ISS)**, mensagens **FWTK (o FireWall ToolKit**, **http://www/fwtk.org**), **wrappers TCP** e mensagens específicas do **Linux**, de modo que já é adequado para uma instalação **Linux** padrão.

LogSentry é escrito em Bourne Shell e C. Ele inclui um utilitário chamado **Logtail** que trata automaticamente da leitura apenas da nova parte dos arquivos de **log**, registrando os números de linha analisados. O sistema é baseado no script **frequentcheck.sh**, escrito por Marcus Ranum e Fred Avolio para o **Firewall Gauntlet**, embora nenhum código seja compartilhado entre eles.

3.2 Ferramenta Logsurfer

Escrito por Wolfgang Ley e Uwe Ellerman no DFN-CERT da Alemanha.

(http://www.cert.dfn.de/eng/logsurf).

É capaz de criar regras dinâmicas e agrupar linhas de **log** em contextos. Enquanto muitas ferramentas operam e geram apenas mensagens de **log** de única linha, o **logsurfer** permite quebrar as mensagens em contextos separados permitindo a análise detalhada.

Se, por exemplo, você viu que alguém conseguiu gravar arquivos em um servidor **FTP** que não deveria ter diretórios com permissão de escrita, provavelmente desejaria determinar quem foi o usuário que gravou arquivos neste diretório.

Como a maior parte do software de verificação de **log**, você teria de ir até o arquivo de **log** original e combinar a linha relatada (a gravação FTP) com o login do usuário, que provavelmente foi ignorada no relatório, pois presume-se que muitas destas linhas estariam presentes.

A configuração do **logsurfer** é um pouco complexa. Ele usa expressões regulares (**regexes** padrão, e nao expressões estendidas em Perl) para

determinar quando uma linha combina com a expressão que voce especifica.

Formato das linhas de configuração:
match-exp not-match-exp stop-exp not-stop-exp timeout action

match-exp : Expressão regular que indica uma combinção e que essa linha deverá ser processada.

not-match-exp : Se o **match-expressão** combinar, mas o **not-match-exp** também combinar, nao a considere como uma combinção (permite a lógica se/mas-não)

stop-exp : Apaga essa regra se a linha combinar com **stop-exp**.

not-stop-exp : Semelhante a not-match-exp, isso significa "apague a regra se **stop-exp** combinar, a menos que **not-stop-exp** também combine"

timeout : Número de segundos em que essa regra devera ser ativa (0 significa tempo sem limite)

action : Uma ação da próxima lista. As acões podem ser seguidas por argumentos opcionais.

Regras permitidas para o campo **'action'** :

ignore : Ignore esta regra

exec : Executa o programa expecificado

pipe: Executa o porgrama especificado e lhe envia a linha de **log** como entrada padrão

open: Inicia um contexto

delete: Apaga um contexto

report: Abre um programa e lhe envia todas as definições de contexto especificadas

rule: Cria uma regra dinâmica

Esta ferramenta oferece mais controle sobre exatamente o que é registrado em **log**, mas é complicado para a configuração e pode consumir muito espaço de memória e CPU do sistema. Por exemplo, embora o padrão para a maioria dos verificadores de **log** seja gerar saída, você precisa chamar explicitamente **/bin/echo** com a opção de **pipe** para realizar qualquer saída do **logsurfer**.

Logsurfer é mais usado para análise de **log** muito específica em conjunto com **LogSentry ou Swatch**, para a verificação de **log** mais profunda.

3.3 Ferramenta Sec

(http://kodu.neti.ee/~risto/sec).

O coordenador de eventos simples, analisa um arquivo, **named pipe** ou entrada padrão. Usando expressões regulares, ele reconhece eventos e pode executar comandos do sistema no caso de uma combinção bem sucedida. Ele pode analisar arquivos de **log**, mas também pode ser integrado a serviços de rede arbitrários, procurando sinais de explorações e realizando comandos quando for apropriado.

Esta ferramenta pode analisar linhas de texto isoladas, várias linhas de texto ou pares de linhas (uma seguida por outra) e pode procurar um limite de linhas que combinam em determinado período de tempo, ignorar certas linhas de texto e realizar ações em horários determinados.

3.4 Ferramenta Lire

(http://logreport.org/lire)

É uma sofisticada ferramenta de análise de **logs** que pode monitorar e criar relatórios de resumo a partir de diversos arquivos de **log** diferentes. Pode ser instalado em seu servidor ou então submeter os **logs** ao **"engine"** de relatório de **Lire** pela internet, recebendo os relatórios de volta por **e-mail**.

O programa **lr_anonymize**, que vem com o pacote **Lire**, torna seus **logs** anônimos, envia para serem processados e depois retira o anonimato dos resultados, quando chegarem por **e-mail.**

Lire possui uma grande lista de formatos de arquivos de log aceitos, incluindo os seguintes:

- Sendmail, Postfix, qmail, exim e nms
- Formato de log comum e combinado do Apache, mod_gzip do Apache
- DNS Bind versoes 8 e 9
- Firewalls: Cisco, ipfilter, ipchains e iptables
- FTP xferlog
- Logs de impressora CUPS e LPRng
- Servidores Proxy Squid e WELF
- Banco de Dados MySQL

Lire converte esses formatos para o Distilled Log Format (DLF), que depois é processado.

Os relatórios são muito úteis tanto para detectar as anomalias como também para ajudá-lo a sintonizar melhor seu sistema e entender suas necessidades específicas.

3.5 Protegendo seus arquivos de log

Se os arquivos de **log** estiverem com as permissões corretas, eles ainda podem ser adulterados.

Se o **Hacker** ou **Cracker** obtiver acesso **roo**t, os arquivos de **log** estão comprometidos.

As permissões normais não impedirão que os arquivos de **log** sejam editados.

Usando permissões específicas de **filesystem**, pode-se impedir que até mesmo o usuário **root** mexa nos arquivos de **log**.

Para os **filesystem ext2 e ext3** o comando que protege os arquivos é o **"chattr"** : **chattr +a /var/log/messages**, coloca o arquivo messages no modo de apenas acréscimo (**append**), o que significa que o cracker não pode mais apagar ou excluir o arquivo, somente acrescentar algo a ele. Isso permite que o processo **syslog** continue enviando novos **logs** ao arquivo, mas nenhum processo poderá mexer nos **logs** antigos.

O **chattr** só impede os usuários mais novatos, os **"script kiddies"**.

Os mais experientes podem usar o comando **chattr** com o atributo **-a**, que remove a permissão de modificações.

No **Sistema Operacional FreeBSD** há o comando **chflags**, que permite trabalhar com os **flags** dos arquivos, tornando-os imutáveis, **append-only e undeletable**. Permite também que estruturas inteiras de diretórios, ou partições, sejam consideradas imutáveis, ou seja, o diretório

/sbin, por exemplo, pode ter todos os seus arquivos apenas para execução, não se consegue adicionar nenhum arquivo neste diretório.

Há outros niveis de segurança mais específicos, chegando até a fazer com que nada seja alterado no Sistema Operacional.

O Sistema Operacional fica totalmente **Read Only**.

O assunto e' bastante extenso e deve ser melhor analisado em cursos relacionados a **Segurança em UNIX.**

3.6 Daemon Syslog-ng

Syslog-ng (**Next generation logging daemon**) (http://www.balabit.com/products/syslog-ng) é um daemon de **logging** do sistema melhor do que **syslogd**, embora normalmente não esteja instalado como o **default**. O arquivo de configuração para **syslog-ng**, chamado **syslog-ng.conf**, é radicalmente diferente de um arquivo **syslog.conf** normal. Assim como **syslog**, você pode especificar vários destinos (arquivos locais, servidores remotos e assim por diante).

Contudo, você também pode definir as origens das mensagens e atuar de modo diferente para gerar eventos localmente **versus** mensagens **syslog** remotas, por exemplo.

Ainda mais poderosa é a capacidade de filtrar mensagens com base em **expressões regulares**, em vez de simplesmente jogar todas as mensagens do **daemon** para um único destino, para analisá-la manualmente.

Syslog-ng pode enviar e receber mensagens **TCP**, além de **UDP**, o que significa que você pode ativar um **syslogging** confiável (**TCP** garante remessa de pacote, enquanto **UDP** não). Apenas por esse motivo, **syslog-ng** pode ser mais útil em ambientes em que você precisa ter certeza de que nenhum **log** será perdido ao enviar seus **logs** para um **host** de **logging** dedicado.

Os manuais de **syslog-ng.conf e syslog-ng** mostram como deve ser os formatos dos arquivos de **log** gerados.

3.7 Cuidados com os arquivos de log

Os arquivos de **log** devem ser protegidos contra alterações. As permissões devem ser de leitura para apenas um usuário e um grupo de usuários.

- Criar um grupo e um usuário para trabalhar com os arquivos de **log** é uma boa medida.
- O diretório **/var/log** não pode ser gravado por nenhum outro usuário além do **root** ou outro criado especialmente para manipular este diretório.
- É recomendável que se faça um **backup** do diretório **/var/log** constantemente. Use o **cron** para estas tarefas.
- Impeça a alteração e a remoção dos arquivos de **log.**
- É possível inserir entradas de **log** falsas nos arquivos de **log**. O comando **logger** permite que isso seja feito.
  $> logger -p facility.level "message"
- Se o **cracker** cirar entradas de **log** semelhantes a essa mensagem de erro, poderá enganar o Administrador de Sistemas para que pense que outro usuário está tentando obter acesso **root**.
  Ele poderia tentar o seguinte:
  $> logger -p kern.alert "authentication failure; logname=public uid=509 guid=0 tty= ruser= rhost= user=root"
  No **Linux Debian,** a linha acima altera o arquivo **/var/log/syslog.** No **FreeBSD** o arquivo adulterado é o **/var/log/messages.**

- Leia os **logs** cuidadosamente. Uma boa análise permite identificar as atividades do invasor, dos **hackers, crackers e outros.**
- Analise o conteúdo atentamente. No exemplo anterior, usando o comando **logger**, parece uma tentativa de **login root** sem sucesso.
- O comando **logger pode ser executado por qualquer usuário no UNIX. Cuidado com falsos alertas.**
- Confie nos seus **logs** assim como pode confiar nos seus usuários. Não chegue a conclusão fácil até que tenha comparado com outra evidência de intrusão.

3.8 Exercícios

1. Instalação e configuração das ferramentas (escolher uma): **Logcheck, Logsurf, Swatch, Sec, Lire, Webalizer, WebLog, Analog.**
2. Personalizar os arquivos de configuração.
3. Configurar a ferramenta escolhida de forma a receber notificações via e-mail.
4. Configurar o **Linux** de modo a exibir os **logs** num terminal TTY.
5. Estabelecer um **loghost** na rede que está sendo utilizada.
6. Instalação e configuração do **syslog-ng**.
7. Instalação e utilização da ferramente **sysstat**. Examinar os recursos relacionados a monitoração de dispositivos de I/O, Processadores, Memória, etc.
8. Elaborar um sistema de **backup** dos arquivos de **log**. Utilize o comando **tar** e coloque o serviço **cron** para efetuar o **backup** periodicamente.
9. Usar o comando **logger** para alterar os arquivos de **log**. Explique os problemas que podem ocorrer na utilização deste recurso.
10. Elabore uma rede com servidores e **loghosts** distribuídos. Explique as configurações que podem ser utilizadas.

3.9 Referências Bibliográficas

1. Segurança contra Hackers Linux - Brian Hatch, James Lee e George Kurtz - Editora Futura - 2003
2. Manual de Administração do Sistema UNIX - Evi Nemeth, Garth Snyder, Scott Seebass, Trent R. Hein - Editora Bookman - 3.ed. - 2002
3. Red Hat Linux Security and Optimization - Mohammed J. Kabir - Wiley Publishing - 2002
4. Linux System Security - The Administrator's Guide to Open Source Security Tools - Scott Mann and Ellen L. Mitchell - Prentice Hall PTR - 2000
5. Segurança Máxima - O guia de um hacker para proteger seu site na Internet e sua rede - Autor anônimo - Editora Campus - 2000
6. Anais dos SSI 2001, SSI 2002 e SSI 2003 - CTA - São José dos Campos - SP

Exemplos de logs em páginas WEB:

- http://www.linorg.cirp.usp.br/webalizer2/index.html
- http://www.linorg.cirp.usp.br/Analog/stats.html

Monitoração de servidores via WEB:

- BigBrother : http://143.107.200.102/bb/
- Nagios : http://143.107.200.102/nagios/

Documentação do Linux Debian e FreeBSD:

- http://www.linorg.cirp.usp.br/Debian.refs/
- http://www.debian.org/doc/
- http://www.linorg.cirp.usp.br/dwww/
- http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/
- http://www.linorg.cirp.usp.br/Manual.FreeBSD/

Revistas:

- http://www.guiatecnico.com.br/EvidenciaDigital
- http://www.gazettelinux.com
- http://www.linuxjournal.com