Sessão 6: Registro de eventos

As atividades 1, 2 e 3 desta sessão serão realizadas na máquina virtual *Client_Linux*. As atividades 4, 5, 6 e 7 serão realizadas em ambas as máquinas *Server_Linux* e *Client_Linux*, de acordo com o enunciado de cada exercício.

Em algumas atividades, você trabalhará com a conta `root`, o que lhe dará todos os direitos sobre os recursos do sistema. Seja cauteloso antes de executar qualquer comando.

1) Registrando os eventos do kernel

1. Configure seu sistema de modo que os eventos gerados pelo kernel sejam registrados em um arquivo chamado `kernel.log`, no diretório `/var/log`.

```
# echo "kern.*        -/var/log/kernel.log" >> /etc/rsyslog.conf
# systemctl restart rsyslog.service

# cat /var/log/kernel.log
cat: /var/log/kernel.log: Arquivo ou diretório não encontrado
```

   Mesmo após reiniciar o *daemon* `rsyslog`, o arquivo não será criado de imediato. Para testar o funcionamento da diretiva, precisamos gerar alguma mensagem para a *facility* apropriada:

```
# modprobe lp
# cat /var/log/kernel.log
Aug  9 11:15:45 cliente kernel: [  447.128333] lp: driver loaded but no devices
found
```

2) Analisando os arquivos de log do sistema

Para esta atividade você terá que ter acesso `ssh` à máquina em que está configurando o sistema de logs para que você possa acompanhar, em tempo real, os registros gravados nos arquivos de log.

1. Crie, em sua máquina, uma conta com senha para acesso via `ssh`.

```
# useradd -m aluno2
# passwd aluno2
Digite a nova senha UNIX:
Redigite a nova senha UNIX:
passwd: senha atualizada com sucesso
```

2. A partir de uma máquina remota, faça login via `ssh` utilizando a conta criada no passo anterior.

Utilize o comando `tail` com a opção `-f` para verificar em tempo real os registros gerados pelo `syslog` no arquivo `/var/log/auth.log`.

No servidor `ssh`, execute:

```
# tail -f -n0 /var/log/auth.log
```

De outra máquina, faça login via `ssh` com a conta criada anteriormente:

```
$ ssh aluno2@192.168.0.20
aluno2@192.168.0.20's password:

aluno2@cliente:~$
```

Monitore o que aconteceu no arquivo `/var/log/auth.log`:

```
# tail -f -n0 /var/log/auth.log
Aug  9 11:26:24 cliente sshd[1050]: Accepted password for aluno2 from 192.168.0.254
port 50325 ssh2
Aug  9 11:26:24 cliente sshd[1050]: pam_unix(sshd:session): session opened for user
aluno2 by (uid=0)
```

3. Faça um *script* que contabilize o número de tentativas de login mal sucedidas através do `ssh`, listando os IPs de origem e quantas tentativas foram feitas por cada IP.

   O *script shell* abaixo mostra um exemplo de solução para o problema proposto:

```
#!/bin/bash

if [[ $EUID -ne 0 ]]; then
  echo "  [*] Not root!" 1>&2
  exit 1
fi

while read -r line; do
  s=( $( echo $line ) )
  echo -e "Host ${s[1]}: ${s[0]} failed logins"
done < <( grep "(sshd.auth): authentication failure.*rhost=" /var/log/auth.log |
awk '{print $14}' | cut -d'=' -f2 | sort -n | uniq -c )
```

3) Analisando os arquivos de log binários do sistema

Nesta atividade, você irá trabalhar com os arquivos de log binários armazenados no diretório `/var/log`.

1. Verifique quais foram os dois últimos usuários a efetuarem login em seu computador.

```
$ last | head -n2
aluno2   pts/1        192.168.0.254     Thu Aug  9 11:26 - 11:27  (00:01)
aluno    pts/0        192.168.0.254     Thu Aug  9 11:10   still logged in
```

2. Como você poderia verificar as contas existentes em seu computador que nunca efetuaram login?

```
$ lastlog | grep '**Nunca logou**' | sort
avahi-autoipd                    **Nunca logou**
backup                           **Nunca logou**
bin                              **Nunca logou**
daemon                           **Nunca logou**
Debian-exim                      **Nunca logou**
funcionario                      **Nunca logou**
games                            **Nunca logou**
gnats                            **Nunca logou**
irc                              **Nunca logou**
list                             **Nunca logou**
lp                               **Nunca logou**
mail                             **Nunca logou**
man                              **Nunca logou**
marcelo                          **Nunca logou**
messagebus                       **Nunca logou**
news                             **Nunca logou**
nobody                           **Nunca logou**
pedro                            **Nunca logou**
proxy                            **Nunca logou**
sshd                             **Nunca logou**
statd                            **Nunca logou**
sync                             **Nunca logou**
sys                              **Nunca logou**
systemd-bus-proxy                 **Nunca logou**
systemd-network                  **Nunca logou**
systemd-resolve                  **Nunca logou**
systemd-timesync                 **Nunca logou**
uucp                             **Nunca logou**
www-data                         **Nunca logou**
```

3. Qual a maneira mais fácil de identificar um login remoto efetuado em seu computador?

   Através do comando `last`. A terceira coluna mostra o *host* de origem do login, seja ele local ou remoto:

```
$ last | head -n20 | grep -v '^reboot'
aluno2   pts/1        192.168.0.254     Thu Aug  9 11:26 - 11:27  (00:01)
aluno    pts/0        192.168.0.254     Thu Aug  9 11:10   still logged in
root     tty1                          Thu Aug  9 03:25 - down   (00:00)
aluno    pts/0        192.168.0.254     Thu Aug  9 02:32 - 03:25  (00:53)
aluno    pts/0        192.168.0.254     Thu Aug  9 02:25 - down   (00:05)
aluno    pts/0        192.168.0.254     Thu Aug  9 01:47 - down   (00:37)
root     tty1                          Wed Aug  8 19:05 - down   (00:00)
aluno    pts/0        192.168.0.254     Wed Aug  8 18:19 - 19:05  (00:46)
root     tty1                          Tue Aug  7 18:18 - down   (00:00)
aluno    pts/0        192.168.0.254     Tue Aug  7 17:56 - 18:17  (00:21)
aluno    pts/1        192.168.0.254     Tue Aug  7 17:07 - 17:15  (00:07)
instruto pts/1        localhost         Tue Aug  7 15:45 - 16:01  (00:15)
instruto pts/1        localhost         Tue Aug  7 14:44 - 14:46  (00:01)
instruto pts/1        localhost         Tue Aug  7 14:42 - 14:42  (00:00)
instruto pts/1        localhost         Tue Aug  7 14:39 - 14:39  (00:00)
```

4. Faça um *script* que mostre o tempo total que cada usuário ficou logado no sistema utilizando as informações obtidas com o comando `last`.

   O *script shell* abaixo mostra um exemplo de solução para o problema proposto:

```
#!/bin/bash

users=( $( last -w | egrep '(tty|pts)' | awk '{print $1}' | sort | uniq ) )

for user in "${users[@]}"; do
  times=( $( last -w | egrep "^$user " | egrep '(tty|pts)' | egrep -v 'still logged
in *$' | sed 's/  *$//' | awk -F '[ :()]' '{printf "%s:%s\n", $(NF-2), $(NF-1)}' )
)

  h=0
  m=0
  for time in "${times[@]}" ; do
    s=( $( echo $time | tr ':' ' ' ) )
    ((h+=${s[0]}))
    ((m+=${s[1]}))
  done

  mh=$(($m/60))
  mr=$(($m%60))
  ((h+=$mh))

  echo "User \"$user\" logged time: $h hours, $mr minutes"
done
```

4) Servidor de log remoto

1. Este exercício deve ser feito utilizando duas máquinas virtuais Linux. Configure um servidor de logs na máquina virtual *Server_Linux*; posteriormente, configure a máquina virtual *Client_Linux* para enviar os registros dos eventos gerados para esse servidor de logs.

   Na máquina *Server_Linux*, edite o arquivo `/etc/rsyslog.conf` e descomente as linhas que se seguem. Em seguida, reinicie o serviço do `rsyslog`.

   ```
   # grep -A1 'imudp' /etc/rsyslog.conf
   $ModLoad imudp
   $UDPServerRun 514

   # systemctl restart rsyslog.service
   ```

   Na máquina *Client_Linux*, configure o envio de logs para o servidor remoto editando o arquivo `/etc/rsyslog.conf` e inserindo a linha que se segue ao final do arquivo, substituindo o endereço IP `192.168.0.10` pelo IP da máquina *Server_Linux*. Em seguida, reinicie o serviço do `rsyslog`.

   ```
   # tail -n1 /etc/rsyslog.conf
   *.*                     @192.168.0.10

   # systemctl restart rsyslog.service
   ```

2. Após terminar a configuração, efetue um login na máquina *Client_Linux* em um terminal qualquer e verifique onde foi registrado esse evento no servidor de logs *Server_Linux*.

   Tendo em vista que o evento gerado na máquina *Client_Linux* será de login, o registro deverá ser enviado para o arquivo onde eventos de autenticação são enviados, na *facility* `authpriv`:

   ```
   # grep '^auth,authpriv' /etc/rsyslog.conf
   auth,authpriv.*                 /var/log/auth.log
   ```

   Sabendo que o arquivo a ser monitorado é o `/var/log/auth.log`, usaremos o comando `tail` para fazê-lo:

   ```
   # tail -f -n0 /var/log/auth.log
   ```

   Após gerar um evento de login via `ssh` na máquina *Client_Linux*, imediamente a mesma mensagem aparece replicada nos logs da máquina *Server_Linux*:

```
# tail -f -n0 /var/log/auth.log
Aug  9 15:18:07 cliente sshd[3285]: Accepted password for aluno from 192.168.0.254
port 50854 ssh2
Aug  9 15:18:07 cliente sshd[3285]: pam_unix(sshd:session): session opened for user
aluno by (uid=0)
```

Evidentemente, é muito confuso ter todas as mensagens de log de uma máquina remota sendo colocadas nos mesmos arquivos que registram os eventos do servidor local. Para tratar esses logs com mais clareza, é interessante separar os logs de cada *host* remoto em seus próprios arquivos e pastas para facilitar o processamento e entendimento. A seguinte configuração pode ser útil para atingir esse objetivo.

Primeiro, note que o `rsyslog` inclui arquivos customizados pelo usuário terminados com a extensão `.conf` no diretório `/etc/rsyslog.d`:

```
# grep '^\$IncludeConfig' /etc/rsyslog.conf
$IncludeConfig /etc/rsyslog.d/*.conf
```

Vamos criar um arquivo novo nessa pasta, `/etc/rsyslog.d/client_linux.conf`, indicando um arquivo específico para envio dos logs da máquina *Client_Linux*, e evitar a escrita desses registros em qualquer outro arquivo local (palavra-chave `stop`). Feito isso, basta reiniciar o *daemon* `rsyslog`. Veja abaixo o conteúdo desse arquivo:

```
if $fromhost-ip == '192.168.0.25'     then /var/log/client_linux.log
& stop
```

Pronto! Agora, novos eventos gerados pela máquina *Client_Linux* serão enviados exclusivamente para o arquivo `/var/log/client_linux.log`, sem se misturar com os eventos locais do servidor de logs.

```
# tail -f -n0 /var/log/client_linux.log
Aug  9 15:34:33 cliente sshd[3340]: Accepted password for aluno from
192.168.0.254 port 50902 ssh2
Aug  9 15:34:33 cliente sshd[3340]: pam_unix(sshd:session): session
opened for user aluno by (uid=0)
```

3. Cite três vantagens obtidas com o uso de um servidor de logs.
   - Facilita o gerenciamento dos arquivos de log, já que estão centralizados em um único servidor.
   - Aumenta a segurança no armazenamento dos arquivos de log, pois o servidor pode estar em outra rede, com regras diferenciadas, dificultando o acesso de possíveis invasores.

- Facilita o backup dos arquivos de log.

5) Utilizando o logger

Nesta atividade, você irá verificar uma funcionalidade importante do comando `logger`.

1. Na máquina *Server_Linux*, inclua uma nova regra no arquivo `/etc/rsyslog.conf`, de modo que qualquer evento gerado pelo daemon `cron` seja registrado no arquivo `/var/log/cron.log`.

```
# tail -n1 /etc/rsyslog.conf
cron.*                  /var/log/cron.log

# systemctl restart rsyslog.service
```

2. Utilize o comando `logger` para testar se a alteração feita no passo anterior produziu o efeito esperado.

```
# logger -p cron.info "teste"

# tail /var/log/cron.log
Aug  9 15:52:26 servidor aluno: teste
```

6) Rotacionando arquivos de log do sistema

Nesta atividade, você irá configurar o rotacionamento dos arquivos de log de seu computador.

1. Na máquina *Server_Linux*, realize o rotacionamento mensal do arquivo recém-criado `/var/log/cron.log`, mantendo uma cópia dos dois últimos arquivos compactados e criando, automaticamente, um novo arquivo vazio após o rotacionamento.

   No arquivo `/etc/logrotate.conf` estão as configurações globais para o rotacionamento dos arquivos de log. Ao configurar o rotacionamento de um arquivo ou um grupo de logs podemos editar diretamente esse arquivo ou, opcionalmente, incluir novas configurações dentro do diretório `/etc/logrotate.d`.

```
# grep '^include' /etc/logrotate.conf
include /etc/logrotate.d

# ls /etc/logrotate.d/
apt  aptitude  dpkg  exim4-base  exim4-paniclog  iptraf  rsyslog
```

   Vamos criar um arquivo `/etc/logrotate.d/cron` para configurar os aspectos de rotacionamento de logs desse arquivo de acordo com os parâmetro especificados no exercício, com o seguinte conteúdo:

```
/var/log/cron.log
{
  rotate 2
  monthly
  missingok
  notifempty
  delaycompress
  compress
  create 640 root adm
  postrotate
    systemctl reload cron.service > /dev/null
  endscript
}
```

7) Aplicativos para análise de arquivos de log

1. Na máquina *Server_Linux*, instale o pacote `logwatch` através do comando `apt-get` e configure-o para enviar um relatório diário do sistema para o usuário `root`. Um exemplo do arquivo de configuração está disponível em `/usr/share/logwatch/default.conf/logwatch.conf`.

   Primeiro, vamos instalar o pacote:

   ```
   # apt-get install logwatch
   ```

   A seguir, vamos copiar o modelo do arquivo de configuração em `/usr/share/logwatch/default.conf/logwatch.conf` para o diretório `/etc/logwatch/conf`:

   ```
   # cp /usr/share/logwatch/default.conf/logwatch.conf /etc/logwatch/conf/
   ```

   Edite o arquivo para que o período e opções de envio fiquem de acordo com o solicitado pela atividade. Abaixo mostramos o conteúdo do arquivo `/etc/logwatch/conf/logwatch.conf`, excluídas linhas de comentário:

   ```
   LogDir = /var/log
   TmpDir = /var/cache/logwatch
   Output = mail
   Format = text
   Encode = none
   MailTo = root
   MailFrom = Logwatch
   Range = All
   Detail = Low
   Service = All
   mailer = "/usr/sbin/sendmail -t"
   ```

Lembre-se de criar o diretório `/var/cache/logwatch`, que ainda não existe:

```
# mkdir /var/cache/logwatch
```

Finalmente, observe que por padrão o Debian já habilita a execução diária do `logwatch` através de um *script* instalado pelo próprio pacote no diretório `/etc/cron.daily`:

```
# ls /etc/cron.daily/ | grep 'logwatch'
00logwatch

# cat /etc/cron.daily/00logwatch | grep -v '^#' | sed '/^$/d'
test -x /usr/share/logwatch/scripts/logwatch.pl || exit 0
/usr/sbin/logwatch --output mail
```

2. Ainda na máquina *Server_Linux*, crie uma regra para o `swatch` que envie um e-mail de notificação ao administrador quando alguma tentativa de login via `ssh`, ou `su` para o usuário `root`, falharem.

   Primeiro, vamos instalar o `swatch` via `apt-get`:

   ```
   # apt-get install swatch
   ```

   A configuração do `swatch` é um tanto quanto arcana, mas a página de manual do programa (`$ man 1p swatch`) nos dá algum direcionamento através da seção *CONFIGURATION EXAMPLE*. Um dos requisitos é criar um arquivo de configuração com a expressão regular que casa com o erro de autenticação do daemon do `sshd`. Primeiro, precisamos conhecer o formato da mensagem:

   ```
   Aug  9 16:39:56 servidor sshd[4113]: Failed password for aluno from 192.168.0.254
   port 51230 ssh2
   ```

   Outro ponto de atenção é a tentativa de `su` para o usuário `root` com falha, possivelmente por senha incorreta. Vamos verificar o formato da mensagem de log:

   ```
   Aug  9 16:46:29 servidor su[4175]: FAILED su for root by aluno
   ```

   Sabendo os formatos objetivados, vamos agora elaborar expressões regulares que casem com os padrões acima, extraiam informação relevante, e executem uma ação apropriada — enviar e-mail de notificação ao usuário `root` em caso de violação desses padrões. Abaixo mostramos o conteúdo do arquivo `/etc/swatch.conf`:

```
watchfor /^(.*sshd\[[0-9]*\]\: Failed password for [A-Za-z0-9]* from ([0-9:.]*).*)/
  exec "echo '$1' | mail root -s '[swatch][ssh]:\ $2' "
  echo

watchfor /^(.*su\[[0-9]*\]\: FAILED su for root by ([A-Za-z0-9]*))/
  exec "echo '$1' | mail root -s '[swatch][su]:\ $2' "
  echo
```

Vamos rodar o `swatch` manualmente e testar se os padrões estão sendo capturados. Serão realizadas duas ações de violação — um login `ssh` com senha incorreta e uma tentativa de `su` para `root` com senha incorreta.

```
# swatch --tail-file=/var/log/auth.log --config-file=/etc/swatch.conf --pid
-file=/var/run/swatch.pid

*** swatch version 3.2.3 (pid:5011) started at Qui Ago  9 17:29:51 -03 2018

Aug  9 17:32:35 servidor sshd[5093]: Failed password for aluno from 192.168.0.254
port 51460 ssh2
Aug  9 17:32:43 servidor su[5117]: FAILED su for root by aluno
```

Aparentemente, tudo funcionou. Vamos verificar se os e-mails estão sendo de fato enviados:

```
$ mail
Mail version 8.1.2 01/15/2001.  Type ? for help.
"/var/mail/aluno": 2 messages 2 new
>N  1 root@servidor.emp  Thu Aug 09 17:32   16/705   [swatch][ssh]: 192.168.0.254
 N  2 root@servidor.emp  Thu Aug 09 17:32   16/663   [swatch][su]: aluno
& 1
Message 1:
From root@servidor.empresa.com.br Thu Aug 09 17:32:35 2018
Envelope-to: root@servidor.empresa.com.br
Delivery-date: Thu, 09 Aug 2018 17:32:35 -0300
To: root@servidor.empresa.com.br
Subject: [swatch][ssh]: 192.168.0.254
From: root <root@servidor.empresa.com.br>
Date: Thu, 09 Aug 2018 17:32:35 -0300

Aug  9 17:32:35 servidor sshd[5093]: Failed password for aluno from 192.168.0.254
port 51460 ssh2

& 2
Message 2:
From root@servidor.empresa.com.br Thu Aug 09 17:32:43 2018
Envelope-to: root@servidor.empresa.com.br
Delivery-date: Thu, 09 Aug 2018 17:32:43 -0300
To: root@servidor.empresa.com.br
Subject: [swatch][su]: aluno
From: root <root@servidor.empresa.com.br>
Date: Thu, 09 Aug 2018 17:32:43 -0300

Aug  9 17:32:43 servidor su[5117]: FAILED su for root by aluno
```

Excelente! Para que o `swatch` não tenha que ser iniciado manualmente, e continue operando mesmo após o reinício do sistema, é necessário que ele possua um *initscript* correspondente. Infelizmente, a versão instalada pelo apt-get não disponibiliza tal facilidade nem em formato legado (no diretório `/etc/init.d`) nem em arquivo de serviço para o `systemd` (que ficam no diretório `/etc/systemd/system`).

Felizmente, é relativamente fácil criar um arquivo de serviço para o `systemd` manualmente. Abaixo mostramos o conteúdo do arquivo `/etc/systemd/system/swatch.service`:

```
[Unit]
Description=Swatch Log Monitoring Daemon
After=syslog.target network.target auditd.service sshd.service

[Service]
ExecStart=/usr/bin/swatch --config-file=/etc/swatch.conf --tail-file
=/var/log/auth.log --pid-file=/var/run/swatch.pid --daemon
ExecStop=/bin/kill -s KILL $(cat /var/run/swatch.pid)
Type=forking
PIDFile=/var/run/swatch.pid

[Install]
WantedBy=multi-user.target
```

Uma vez criado, deve-se instruir o `systemd` a carregar o arquivo:

```
# systemctl daemon-reload
```

Pronto! Agora é possível habilitar/desabilitar o `swatch` durante o *boot* do sistema, e iniciar/parar/reiniciar e verificar o estado do serviço normalmente:

```
# systemctl enable swatch.service
Created symlink from /etc/systemd/system/multi-user.target.wants/swatch.service to
/etc/systemd/system/swatch.service.

# systemctl is-enabled swatch.service
enabled

# systemctl start swatch.service

# systemctl status swatch.service
● swatch.service - Swatch Log Monitoring Daemon
   Loaded: loaded (/etc/systemd/system/swatch.service; enabled)
   Active: active (running) since Qui 2018-08-09 17:37:57 -03; 4s ago
  Process: 5216 ExecStart=/usr/bin/swatch --config-file=/etc/swatch.conf --tail
-file=/var/log/auth.log --pid-file=/var/run/swatch.pid --daemon (code=exited,
status=0/SUCCESS)
 Main PID: 5218 (/usr/bin/swatch)
   CGroup: /system.slice/swatch.service
           ├─5218 /usr/bin/swatch --config-file=/etc/swatch.conf --tail
-file=/var/log/auth...
           └─5219 /usr/bin/tail -n 0 -F /var/log/auth.log

Ago 09 17:37:57 servidor systemd[1]: Starting Swatch Log Monitoring Daemon...
Ago 09 17:37:57 servidor systemd[1]: PID file /var/run/swatch.pid not readable
(yet?) a...rt.
Ago 09 17:37:57 servidor systemd[1]: Started Swatch Log Monitoring Daemon.
Hint: Some lines were ellipsized, use -l to show in full.
```

3. Ainda na máquina *Server_Linux*, habilite o `logcheck` para enviar relatórios ao usuário `root` de 30 em 30 minutos (ex: 1:00, 1:30, etc.).

   Primeiro, vamos instalar o `logcheck` via `apt-get`:

   ```
   # apt-get install logcheck
   ```

   O `logcheck` já vem com envio de e-mails habilitado por padrão, então a única configuração necessária é alterar a periodicidade de envio de relatórios. O arquivo `/etc/cron.d/logcheck` vem configurado para envios de hora em hora. Edite a linha:

   ```
   2 * * * *       logcheck    if [ -x /usr/sbin/logcheck ]; then nice -n10
   /usr/sbin/logcheck; fi
   ```

   Alterando-a para:

   ```
   0,30 * * * *       logcheck    if [ -x /usr/sbin/logcheck ]; then nice -n10
   /usr/sbin/logcheck; fi
   ```

O `logcheck` fará um *scan* dos logs de sistema e enviará por e-mail linhas consideradas "interessantes" — note que o programa envia apenas os registros ocorridos desde a sua última execução.

8) Recomendações básicas de segurança

1. O que você faria para aumentar o nível de segurança em um servidor de logs centralizado? Cite duas opções.
    - Desabilitar o serviço `sshd` no servidor de logs, permitindo acesso somente pela console.
    - Configurar o firewall de *host* para permitir apenas tráfego de pacotes UDP na porta 514.
    - Utilizar uma rede isolada para a troca de mensagens de log.
    - Desinstalar todos os serviços que não estão sendo utilizados ou são desnecessários à função do servidor.
    - Manter o sistema operacional rigorosamente atualizado.