Controle de anomalias e bloqueio de ataques em redes em tempo real

João Eriberto Mota Filho

Brasília, DF, 29 abr. 2020

Sumário

Sumário

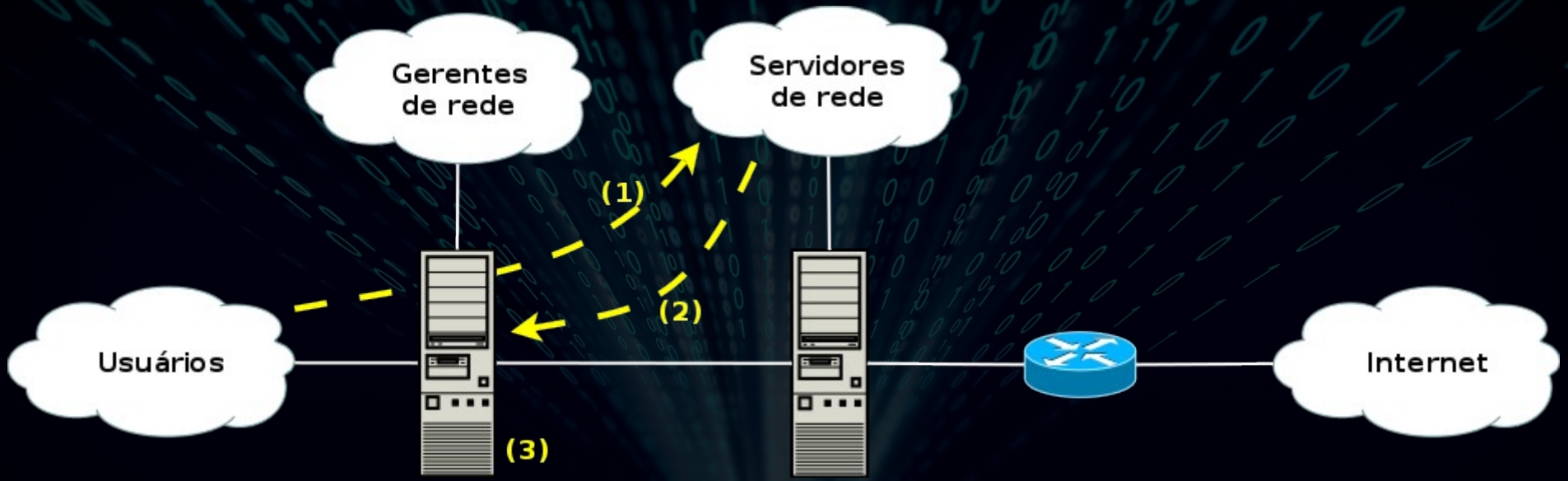
Visão geral do processo

- A ideia é gerar alertas de anomalias, via log, e tratá-los imediatamente.
- Esses alertas poderão ser processados local ou remotamente.
- Com isso, será possível atuar, de forma personalizada, sobre eventos em toda a rede.
- Tudo o que você precisa fazer é gerar logs!
- Esses logs serão fontes para bloqueios e alertas.
- Tudo isso em tempo real!

Visão geral do processo

- **Alguns exemplos de ações e alertas, em tempo real, que poderão ocorrer:**
  - ☞ **Bloqueio de máquinas MS Windows na rede.**
  - ☞ **Bloqueio de máquinas realizando scaneamentos de portas (nmap e outros).**
  - ☞ **Bloqueio de máquinas executando scaneamento de vulnerabilidades (OpenVAS, nikto etc).**
  - ☞ **Bloqueio de máquinas com múltiplos erros de autenticação.**
  - ☞ **Avisos sobre ocorrências de port security em switches Cisco.**

Visão geral do processo



(1) Usuário tenta atacar servidor interno.

(2) Sevidor envia um log de alerta, contendo detalhes do ataque, para uma máquina intermediária com filtro de pacotes.

(3) Máquina intermediária bloqueia IP do atacante, em tempo real, por 1 hora, avisando ao admin sobre o fato.

Sumário

Conceitos básicos e ferramentas

- **Syslog: esse é o principal sistema de logs no GNU/Linux.**
- **O Debian utiliza o servidor rsyslog para prover o syslog.**
- **O rsyslog é amplamente configurável e permite a coleta de dados locais e remotos para gerar os logs.**
- **É possível criar regras, inclusive com expressões regulares, para tratar os logs gerados pelo rsyslog.**

Conceitos básicos e ferramentas

- **inotify: monitor de eventos de filesystem, baseado em informações de inodes.**
- **Os inodes controlam os arquivos e diretórios existentes em um filesystem.**
- **Filesystems são estabelecidos por formatação de disco.**
- **O inotify é parte integrante do Kernel Linux e o seu trabalho é obrigatório para que o SO funcione.**
- **Em resumo, o inotify monitora mudanças nos arquivos e diretórios. Então, poderemos acompanhar atividades em logs!!!**

Conceitos básicos e ferramentas

- **Ferramentas e sistemas para logs:**
  - ☞ **rsyslog:** servidor de logs padrão no Debian e outros.
  - ☞ **logger:** envia logs personalizados para rsyslogs locais ou remotos.
  - ☞ **iptables:** ferramenta do Netfilter que gera logs de rede e faz bloqueios.
  - ☞ **snort:** pode analizar payloads de rede e gerar logs.
  - ☞ **suricata:** faz o mesmo trabalho do snort. (melhor?)
  - ☞ **portsentry:** gera logs sobre port scans e os bloqueia.
  - ☞ **curl:** acessa servidores web, gerando logs, podendo estes serem personalizados.

Conceitos básicos e ferramentas

- **Ferramentas baseadas em inotify:**
  - ☞ **fail2ban:** analisa logs em tempo real e faz bloqueios por tempos predeterminados.
  - ☞ **iwatch:** reage, em tempo real, a alterações em filesystems. Pode ser utilizado para verificar logs.
- **Ferramentas para comunicação em linha de comando:**
  - ☞ **sendxmpp:** envia mensagens XMPP (Jabber).
  - ☞ **sendmail:** envia mensagens de email.
  - ☞ **yowsup-cli:** envia mensagens WhatsApp.
  - ☞ **tg (by vysheng):** envia mensagens Telegram (ver questões de privacidade no bug Debian #737563).

Conceitos básicos e ferramentas

- **Ferramentas e sistemas XMPP (Jabber):**
  - ☞ **prosody:** servidor XMPP com diversos recursos, permitindo criptografia e salas de conferência.
  - ☞ **pidgin:** cliente para desktops. Extremamente versátil. Disponível para Linux, Windows e Mac.
  - ☞ **Yaxim:** cliente leve para Android. Mantém-se, por default, em segundo plano, da mesma forma como faz o WhatsApp. Permite criptografia. Ótimo.
  - ☞ **Freelab:** cliente para Android, com segundo plano e criptografia configuráveis (não é o default).
  - ☞ **IOS?**

Sumário

Geração de logs

- O rsyslog pode ser configurado para receber logs remotos. A porta default é a 514 UDP.
- Para isso, no Debian, basta descomentar as seguintes linhas em /etc/rsyslog.conf:

```
# provides UDP syslog reception
module(load="imudp")
input(type="imudp" port="514")
```

- É uma boa ideia utilizar o iptables para dizer quem poderá acessar a porta 514 UDP do servidor.

Geração de logs

- Gerar log local com logger (pode ser via usuário comum do sistema):

```
$ logger Teste de log
# tail -n1 /var/log/syslog
Oct 17 14:26:14 octans eriberto: Teste de log
```

- Gerar log com ID de processo e TAG:

```
$ logger -i -t TESTE Teste de log
# tail -n1 /var/log/syslog
Oct 17 14:41:21 octans TESTE[32146]: Teste de log
```

Geração de logs

- **Gerar log em servidor remoto (-d = UDP):**

```
$ logger -n logserver.com.ex -d -P 514 -i -t TESTE Teste de log

---> Resultado no servidor remoto (observe o IP):
Oct 17 14:47:25 10.0.0.1 TESTE[32240]: Teste de log
```

- **Se mais de um nome for colocado na TAG, o primeiro será interpretado como sendo o nome do host:**

```
$ logger -n logserver.com.ex -d -P 514 -i -t 'octans TESTE' Teste de log

---> Resultado no servidor remoto (observe o nome):
Oct 17 14:54:13 octans TESTE[32329]: Teste de log
```

Geração de logs

- O rsyslog permite filtragens, fazendo com que cada tipo de log caia em um arquivo de log diferente do /var/log/syslog, que é o default.
- Os arquivos de configuração de filtros deverão ter a extensão .conf e ser colocados em /etc/rsyslog.d/. É possível ter todas as regras em um único arquivo.
- Como exemplo, a linha a seguir enviará todas as mensagens que contiverem a TAG ALERTA para /var/log/01_SERVERS/ALERTA.log:

```
:syslogtag, contains, "ALERTA" /var/log/01_SERVERS/ALERTA.log
```

Geração de logs

- **Composição básica uma linha de log:**

```
Oct 17 14:54:13 octans TESTE[32329]: Teste de log
```

```
TIMESTAMP HOSTNAME programname syslogtag msg
```

- **Dependendo do log, haverá outros campos possíveis. Para mais detalhes, $ man rsyslog.conf.**
- **Dentro de um arquivo, as regras serão processadas na ordem em que aparecerem.**

Geração de logs

- **Outros exemplos de filtros:**

```
# Mensagens oriundas de 10.0.0.1 irão para /var/log/01_SERVERS/01-CISCO.log
:fromhost-ip, contains, "10.0.0.1" /var/log/01_SERVERS/01-CISCO.log
&stop

# Endereços privados (RCF1918) saindo para Internet. Mensagens contendo
# "RFC1918" irão para /var/log/01_SERVERS/ALERTA.log
:msg, contains, "RFC1918" /var/log/01_SERVERS/ALERTA.log
&stop

# Regra com expressão regular
:msg, regex, "fatal .* error" /var/log/01_SERVERS/ALERTA.log

# Separar logs por nome de servidor usando template
$template Servers, "/var/log/01_SERVERS/%HOSTNAME%/%PROGRAMNAME%.log"
*.*   ?Servers
& stop

# REFERÊNCIA: http://www.rsyslog.com/doc/v8-stable/configuration/filters.html
```

Geração de logs

- **Há diversas outras formas de se gerar logs.**
- **Uma possibilidade é enviar dados para um servidor web remoto com o comando curl. Exemplo:**

```
$ curl -so /dev/null 'http://10.0.0.1?ip=10.0.0.52;user=marco'

---> Exemplo de log gerado na máquina alvo:
# tail -n1 /var/log/apache2/access.log
10.0.0.1 - - [21/Oct/2016:01:22:25 -0200] "GET /?ip=10.0.0.52;
user=marco HTTP/1.1" 200 10956 "-" "curl/7.38.0"
```

Sumário

Análise de logs e ações

- Logs poderão ser analizados por vários programas.
- iWatch e fail2ban são duas boas possibilidades baseadas em inotify.
- O fail2ban tem a vantagem de permitir bloqueios por tempos predeterminados.
- Para utilizar o fail2ban é necessário configurar os seguintes arquivos, em /etc/fail2ban/:
  - ☞ action.d/<nome>.conf: ação a ser executada.
  - ☞ filter.d/<nome>.conf: filtro para ocorrências de log.
  - ☞ jail.d/<nome>.conf: determina as condições para realizar a filtragem e desencadear a ação.

Análise de logs e ações

- **Exemplo: action.d/action-route-xmpp.conf:**

```
[Definition]
actionban = ip route add <blocktype> <ip>
            echo `date "+%%d %%b %%Y %%X"` <ip> <name> | \
            { sendxmpp -t eriberto@jabber.org fulano@jabber.org; \
              logger -t fail2ban-block <ip> <name>; }

actionunban = ip route del <blocktype> <ip>

actionstart =
actionstop =
actioncheck =

[Init]
blocktype = unreachable
```

- **<ip> corresponde ao IP que está trafegando; <name> é uma variável que será recebida do jail.conf.**

Análise de logs e ações

- **Considerando a seguinte linha de log recebida pelo servidor web de monitoramento:**

```
10.0.0.1 - - [21/Oct/2016:01:22:25 -0200] "GET /?ip=10.0.0.52;
evento=nikto HTTP/1.1" 200 10956 "-" "curl/7.38.0"
```

- **Exemplo: filter.d/filter-nikto.conf:**

```
[Definition]

failregex = GET .*ip=<HOST>;evento=nikto.*

ignoreregex =
```

Análise de logs e ações

- **Exemplo: jail.d/nikto.conf:**

```
[nikto]
enabled = true
filter  = filter-nikto
logpath = /var/log/apache2/access.log
action  = action-route-xmpp[name=nikto]
maxretry = 2
findtime = 120
bantime = 86400
```

- **Os arquivos do jail.d/ determinam a filtragem e desencadeiam a ação sob certas condições.**

Análise de logs e ações

- O iWatch também poderá ser utilizado para observar a movimentação de logs e, em seguida, executar scripts.
- Outra possibilidade é o uso de um servidor de logs, centralizado, analizando toda a rede em um único ponto central.
- No caso anterior, programas especializados em análise de logs também poderão ser utilizados.
- iptables, snort e suricata são ferramentas eficientes para gerar logs específicos!

Análise de logs e ações

- No caso do snort e do suricata, apague todas as regras que não forem necessárias e adicione as suas. Ainda, se possível, execute-os sobre bridge.
- Exemplo de filtragem com iptables para detectar port scan da rede de usuários comuns (ligada na eth0) para a rede de servidores (192.168.10.0/24):

```
# iptables -A FORWARD -i eth0 -d 192.168.10.0/24 -p tcp -m multiport --dports 22,23,3306 -j LOG --log-prefix 'SCAN REDE '
```

- No caso anterior, deverá haver um filtro de rsyslog desviando tudo que contiver a TAG “SCAN REDE” para um log separado que será monitorado.

Análise de logs e ações

- **Exemplo de log de switch Cisco:**

```
   Sep 22 16:24:03 10.0.0.100 8251: sw_rachel: 4w2d: %LINEPROTO-5-
UPDOWN: Line protocol on Interface GigabitEthernet2/0/33, changed
state to down

   Sep 22 16:24:05 10.0.0.100 8252: sw_rachel: 4w2d: %LINEPROTO-5-
UPDOWN: Line protocol on Interface GigabitEthernet2/0/33, changed
state to up
   Sep 22 16:27:54 10.0.0.100 8254: sw_rachel: 4w2d: %PM-4-
ERR_DISABLE: psecure-violation error detected on Gi1/0/36, putting
Gi1/0/36 in err-disable state
   Sep 22 16:27:54 10.0.0.100 8255: sw_rachel: 4w2d:
%PORT_SECURITY-2-PSECURE_VIOLATION: Security violation occurred,
caused by MAC address 782b.cbc0.ed9d on port
GigabitEthernet1/0/36.
```

- **É possível saber, via Jabber ou outro meio, em tempo real, sobre bloqueios realizados. Use iWatch!**

Análise de logs e ações

- **Filtrando uma possível existência de MS Windows na rede:**

```
# iptables -A FORWARD ! -i eth0 -m ttl --ttl-eq 128 -j LOG --
  log-prefix 'TTL WINDOWS '

# iptables -A FORWARD ! -i eth0 -m ttl --ttl-eq 127 -j LOG --
  log-prefix 'TTL WINDOWS '

# iptables -A FORWARD ! -i eth0 -m ttl --ttl-eq 126 -j LOG --
  log-prefix 'TTL WINDOWS '
```

- **É uma boa ideia estudar os módulos do Netfilter que podem ser ativados pelo iptables ($ man iptables-extensions).**

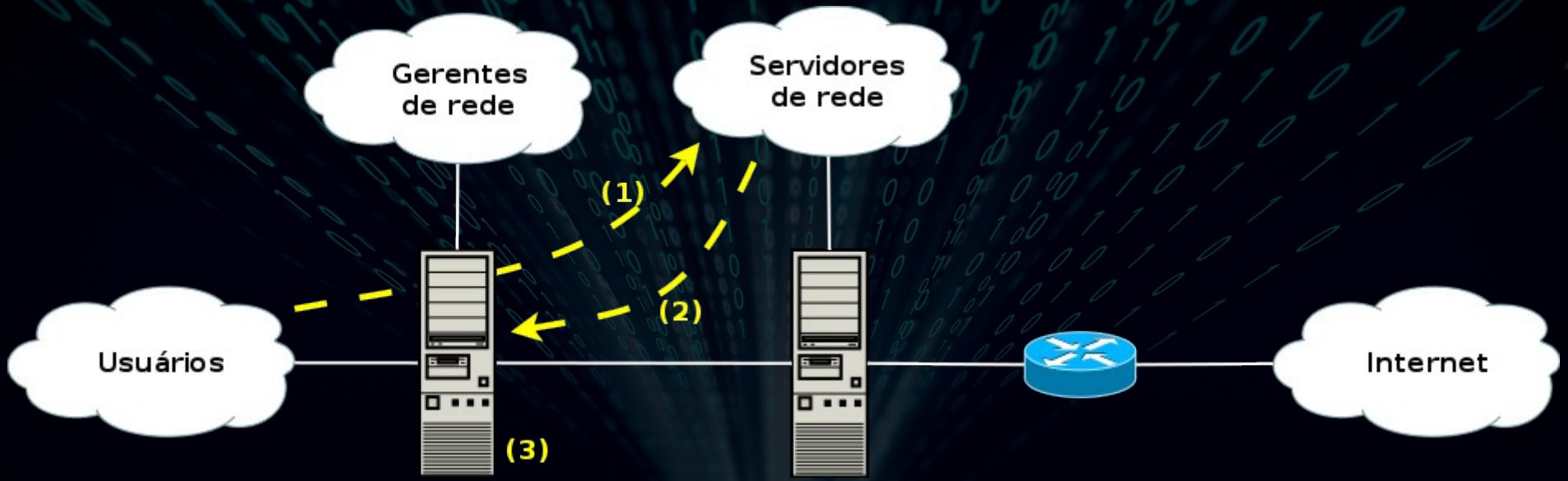
Análise de logs e ações

- A detecção de máquinas Windows poderá ser feita por outros métodos, como a observação de chamadas a servidores atualização na Microsoft ou pela tentativa de conexão a servidores WINS fornecidos via DHCP mas não existentes na rede etc.
- Outro elemento interessante para filtrar port scans, principalmente a partir de ataques oriundos da Internet, é o Portsentry.
- O Portsentry realiza bloqueios mas não os desfaz. A solução para isso é uma combinação de Portsentry, gerando apenas logs, com Fail2Ban atuando.

Análise de logs e ações

- Tentativas de autenticação por força bruta também podem ser facilmente identificados e tratados. Basta observar a quantidade de ocorrências por minuto (ou segundo) nos logs.
- Excessivos erros 404 em curto período de tempo também caracterizam ataques.
- Logs do snort ou do suricata com múltiplos ataques por segundo representam outra boa fonte de filtragem.
- Use a imaginação!

Análise de logs e ações



(1) Usuário tenta atacar servidor interno.

(2) No servidor, um agente detecta a ação localmente e escreve isso em log remoto com logger. O iWatch pode ser utilizado.

(3) Máquina intermediária usa Fail2Ban para bloquear IP do atacante por 1 hora e avisar ao admin via Jabber.

Sumário

Conclusão

- **Com o princípio de processamento de eventos em logs locais ou remotos, é possível tratar diversos tipos de anomalias em redes de forma distribuída.**
- **Vários outros programas, além dos citados nesta palestra, poderão ser utilizados para gerar logs e bloquear ataques.**
- **Use a criatividade! Nenhum sistema de firewall comercial poderá fazer o mesmo que você!**

Esta palestra está disponível em:

http://www.eriberto.pro.br

Siga-me em http://twitter.com/eribertomota