Detectar y bloquear escaneos: portsentry

Por Paco Aldarias Raya

Impreso: 6 de mayo de 2004

Email: pacolinux arroba inicia punto es
Web: `http://pagina.de/pacodebian`
Con Linux Debian. En Valencia (España)

Realizado con: LaTeX

Índice

1. Introducción

Portsentry detecta y bloquea las máquinas que escanean la nuestra.
Dicho en ingles es un "Portscan detection daemon".
Porsentry se pone en marcha como demonio.

2. Instalación

```
apt-get install portsentry
```

Paquetes q instalados:

```
dpkg -l | grep portsentry
ii  portsentry     1.2-4          Portscan detection daemon
```

3. Ficheros de configuración: /etc/portsentry/-portsentry.ignore

Fichero de ips que no bloqueará. Son las ips de confianza

4. Ficheros de configuración: /etc/portsentry/-portsentry.conf

Debemos revisarlo y adaptarlo a nuestro gusto.
Está muy bien explicado.
Un ejemplo se puede ver aqui:
`portsentry.conf`

```
# PortSentry Configuration
#
# $Id: portsentry.conf.Debian,v 1.6 2001/07/19 21:02:20 agx Exp $
#
# Original portsentry.conf by Craig H. Rowland <crowland@psionic.com>
# modified for Debian by Guido Guenther <agx@debian.org>
#
# IMPORTANT NOTE: You CAN NOT put spaces between your port arguments.
#
# The default ports will catch a large number of common probes
#
# All entries must be in quotes.


##########################
# Port Configurations #
##########################
#
```

```
#
# Some example port configs for classic and basic Stealth modes
#
# I like to always keep some ports at the "low" end of the spectrum.
# This will detect a sequential port sweep really quickly and usually
# these ports are not in use (i.e. tcpmux port 1)
#
# ** X-Windows Users **: If you are running X on your box, you need to
# you are not binding PortSentry to port 6000 (or port 2000 for OpenWi
# Doing so will prevent the X-client from starting properly.
#
# These port bindings are *ignored* for Advanced Stealth Scan Detectio
#

# Un-comment these if you are really anal:
#TCP_PORTS="1,7,9,11,15,70,79,80,109,110,111,119,138,139,143,512,513,51
#UDP_PORTS="1,7,9,66,67,68,69,111,137,138,161,162,474,513,517,518,635,6
#
# Use these if you just want to be aware:
TCP_PORTS="1,11,15,79,540,635,1080,1524,2000,5742,6667,12345,12346,200
UDP_PORTS="1,7,9,69,161,162,513,635,640,641,700,37444,34555,31335,3277
#
# Use these for just bare-bones
#TCP_PORTS="1,11,15,110,111,143,540,635,1080,1524,2000,12345,12346,200
#UDP_PORTS="1,7,9,69,161,162,513,640,700,32770,32771,32772,32773,32774

##################################################
# Advanced Stealth Scan Detection Options #
##################################################
#
# This is the number of ports you want PortSentry to monitor in Advanc
# Any port *below* this number will be monitored. Right now it watches
# everything below 1024.
#
# On many Linux systems you cannot bind above port 61000. This is beca
# these ports are used as part of IP masquerading. I don't recommend y
# bind over this number of ports. Realistically: I DON'T RECOMMEND YOU
# OVER 1024 PORTS AS YOUR FALSE ALARM RATE WILL ALMOST CERTAINLY RISE.
# warned! Don't write me if you have have a problem because I'll only
# you to RTFM and don't run above the first 1024 ports.
#
```

```
#
ADVANCED_PORTS_TCP="1024"
ADVANCED_PORTS_UDP="1024"
#
# This field tells PortSentry what ports (besides listening daemons) t
# ignore. This is helpful for services like ident that services such
# as FTP, SMTP, and wrappers look for but you may not run (and probabl
# *shouldn't* IMHO).
#
# By specifying ports here PortSentry will simply not respond to
# incoming requests, in effect PortSentry treats them as if they are
# actual bound daemons. The default ports are ones reported as
# problematic false alarms and should probably be left alone for
# all but the most isolated systems/networks.
#
# Default TCP ident and NetBIOS service
ADVANCED_EXCLUDE_TCP="113,139,9999"
# Default UDP route (RIP), NetBIOS, bootp broadcasts.
ADVANCED_EXCLUDE_UDP="520,138,137,67"


##########################
# Configuration Files#
##########################
#
# Hosts to ignore
IGNORE_FILE="/etc/portsentry/portsentry.ignore"
# Hosts that have been denied (running history)
HISTORY_FILE="/var/lib/portsentry/portsentry.history"
# Hosts that have been denied this session only (temporary until next
BLOCKED_FILE="/var/lib/portsentry/portsentry.blocked"

###################################
# Misc. Configuration Options#
###################################
#
# DNS Name resolution - Setting this to "1" will turn on DNS lookups
# for attacking hosts. Setting it to "0" (or any other value) will shu
# it off.
RESOLVE_HOST = "0"
```

```
####################
# Response Options#
####################
# Options to dispose of attacker. Each is an action that will
# be run if an attack is detected. If you don't want a particular
# option then comment it out and it will be skipped.
#
# The variable $TARGET$ will be substituted with the target attacking
# host when an attack is detected. The variable $PORT$ will be substitu
# with the port that was scanned.
#
####################
# Ignore Options #
####################
# These options allow you to enable automatic response
# options for UDP/TCP. This is useful if you just want
# warnings for connections, but don't want to react for
# a particular protocol (i.e. you want to block TCP, but
# not UDP). To prevent a possible Denial of service attack
# against UDP and stealth scan detection for TCP, you may
# want to disable blocking, but leave the warning enabled.
# I personally would wait for this to become a problem before
# doing though as most attackers really aren't doing this.
# The third option allows you to run just the external command
# in case of a scan to have a pager script or such execute
# but not drop the route. This may be useful for some admins
# who want to block TCP, but only want pager/e-mail warnings
# on UDP, etc.
#
#
# 0 = Do not block UDP/TCP scans.
# 1 = Block UDP/TCP scans.
# 2 = Run external command only (KILL_RUN_CMD)

BLOCK_UDP="1"
BLOCK_TCP="1"

####################
# Dropping Routes:#
####################
# This command is used to drop the route or add the host into
```

```
# a local filter table.
#
# The gateway (333.444.555.666) should ideally be a dead host on
# the *local* subnet. On some hosts you can also point this at
# localhost (127.0.0.1) and get the same effect. NOTE THAT
# 333.444.555.66 WILL *NOT* WORK. YOU NEED TO CHANGE IT!!
#
# ALL KILL ROUTE OPTIONS ARE COMMENTED OUT INITIALLY. Make sure you
# uncomment the correct line for your OS. If you OS is not listed
# here and you have a route drop command that works then please
# mail it to me so I can include it. ONLY ONE KILL_ROUTE OPTION
# CAN BE USED AT A TIME SO DON'T UNCOMMENT MULTIPLE LINES.
#
# NOTE: The route commands are the least optimal way of blocking
# and do not provide complete protection against UDP attacks and
# will still generate alarms for both UDP and stealth scans. I
# always recommend you use a packet filter because they are made
# for this purpose.
#

# Generic
#KILL_ROUTE="/sbin/route add $TARGET$ 333.444.555.666"

# Generic Linux
#KILL_ROUTE="/sbin/route add -host $TARGET$ gw 333.444.555.666"

# Newer versions of Linux support the reject flag now. This
# is cleaner than the above option.
KILL_ROUTE="/sbin/route add -host $TARGET$ reject"

# Generic BSD (BSDI, OpenBSD, NetBSD, FreeBSD)
#KILL_ROUTE="/sbin/route add $TARGET$ 333.444.555.666"

# Generic Sun
#KILL_ROUTE="/usr/sbin/route add $TARGET$ 333.444.555.666 1"

# NEXTSTEP
#KILL_ROUTE="/usr/etc/route add $TARGET$ 127.0.0.1 1"

# FreeBSD
#KILL_ROUTE="route add -net $TARGET$ -netmask 255.255.255.255 127.0.0.1
```

```
# Digital UNIX 4.0D (OSF/1 / Compaq Tru64 UNIX)
#KILL_ROUTE="/sbin/route add -host -blackhole $TARGET$ 127.0.0.1"

# Generic HP-UX
#KILL_ROUTE="/usr/sbin/route add net $TARGET$ netmask 255.255.255.0 12

##
# Using a packet filter is the PREFERRED. The below lines
# work well on many OS's. Remember, you can only uncomment *one*
# KILL_ROUTE option.
##

# ipfwadm support for Linux
#KILL_ROUTE="/sbin/ipfwadm -I -i deny -S $TARGET$ -o"
#
# ipfwadm support for Linux (no logging of denied packets)
#KILL_ROUTE="/sbin/ipfwadm -I -i deny -S $TARGET$"
#
# ipchain support for Linux
#KILL_ROUTE="/sbin/ipchains -I input -s $TARGET$ -j DENY -l"
#
# ipchain support for Linux (no logging of denied packets)
#KILL_ROUTE="/sbin/ipchains -I input -s $TARGET$ -j DENY"
#
# iptables support for Linux
#KILL_ROUTE="/sbin/iptables -I INPUT -s $TARGET$ -j DROP"
#
# iptables support for Linux with limit and LOG support. Logs only
# a limited number of packets to avoid a denial of service attack.
# KILL_ROUTE="/sbin/iptables -I INPUT -s $TARGET$ -j DROP && /sbin/ipt
#
# For those of you running FreeBSD (and compatible) you can
# use their built in firewalling as well.
#
#KILL_ROUTE="/sbin/ipfw add 1 deny all from $TARGET$:255.255.255.255 t
#
#
# For those running ipfilt (OpenBSD, etc.)
# NOTE THAT YOU NEED TO CHANGE external_interface TO A VALID INTERFACE
#
```

```
#KILL_ROUTE="/bin/echo 'block in log on external_interface from $TARGE


################
# TCP Wrappers#
################
# This text will be dropped into the hosts.deny file for wrappers
# to use. There are two formats for TCP wrappers:
#
# Format One: Old Style - The default when extended host processing
# options are not enabled.
#
#KILL_HOSTS_DENY="ALL: $TARGET$"

# Format Two: New Style - The format used when extended option
# processing is enabled. You can drop in extended processing
# options, but be sure you escape all '%' symbols with a backslash
# to prevent problems writing out (i.e. \%c \%h )
#
KILL_HOSTS_DENY="ALL: $TARGET$ : DENY"

####################
# External Command#
####################
# This is a command that is run when a host connects, it can be whatev
# you want it to be (pager, etc.). This command is executed before the
# route is dropped or after depending on the KILL_RUN_CMD_FIRST option
#
#
# I NEVER RECOMMEND YOU PUT IN RETALIATORY ACTIONS AGAINST THE HOST SCA
# YOU!
#
# TCP/IP is an *unauthenticated protocol* and people can make scans ap
# of thin air. The only time it is reasonably safe (and I *never* thin
# reasonable) to run reverse probe scripts is when using the "classic"
# This mode requires a full connect and is very hard to spoof.
#
# The KILL_RUN_CMD_FIRST value should be set to "1" to force the comma
# to run *before* the blocking occurs and should be set to "0" to make
# command run *after* the blocking has occurred.
#
```

```
#KILL_RUN_CMD_FIRST = "0"
#
#
#KILL_RUN_CMD="/some/path/here/script $TARGET$ $PORT$ $MODE$"
# for examples see /usr/share/doc/portsentry/expamples/

KILL_RUN_CMD="/root/portsentry.sh $TARGET$"

##########################
# Scan trigger value#
##########################
# Enter in the number of port connects you will allow before an
# alarm is given. The default is 0 which will react immediately.
# A value of 1 or 2 will reduce false alarms. Anything higher is
# probably not necessary. This value must always be specified, but
# generally can be left at 0.
#
# NOTE: If you are using the advanced detection option you need to
# be careful that you don't make a hair trigger situation. Because
# Advanced mode will react for *any* host connecting to a non-used
# port below your specified range, you have the opportunity to
# really break things. (i.e someone innocently tries to connect to
# you via SSL [TCP port 443] and you immediately block them). Some
# of you may even want this though. Just be careful.
#
SCAN_TRIGGER="0"

##########################
# Port Banner Section#
##########################
#
# Enter text in here you want displayed to a person tripping the PortS
# I *don't* recommend taunting the person as this will aggravate them.
# Leave this commented out to disable the feature
#
# Stealth scan detection modes don't use this feature
#
#PORT_BANNER="** UNAUTHORIZED ACCESS PROHIBITED *** YOUR CONNECTION AT

# EOF
```

Cada vez q se cambia este fichero deberemos reiniciarlo:

```
/etc/init.d/portsentry restart
```

5. Script portsentry.sh

Dentro de /etc/portsentry/portsentry.conf tengo puesto esto:

```
KILL_RUN_CMD="/root/portsentry.sh $TARGET$"
```

Ejecuta el script portsentry.sh

```
d1=/var/log/portsentry/$1.txt
d2=/var/log/portsentryh.txt
d=/var/log/portsentry.txt
lee 'Ataque Ataque Ataque por '  $1 'ya le mano un email'

# Sino se ha escaneado antes
#if  [ ! -d $d1 ];  then
echo '*********************************' >> $d1
echo $1 - $(date +%d-%m-%Y-%H:%M) >> $d2
echo $1 >> $d
echo $(date +%d-%m-%Y-%H:%M) >> $d1
echo $1 >> $d1
echo $1 >> /root/intrusos.txt
nslookup $1 >> $d1
whois $1 >> $d1
echo "Puertos abiertos: " >> $d1
nmap -P0 --max_rtt_timeout 20000 $1 >> $d1
echo "Por horas: /var/log/portsentryh.txt" >> $d1
echo "Solo IPS:  /var/log/portsentry.txt" >> $d1
echo "Intrusos IPS: /root/intrusos.txt" >> $d1
echo "Sistema: " >> $d1
#fi

mail -s "Ataque de $1" paco < $d1
#/root/mamon.sh $1
#/root/flood.sh $1
```

6. Script lee

Es llamado por portsentry.sh. Lee un texto.

```
echo "$CABECERA $1  " |festival --tts --language spanish
```

Índice alfabético