

# **I**NTELLIGENT **DE**FENCE **A**CTIVATION

# Traditional Attack Defence

**STEP #1.** Check the log files (/var/log/secure) periodically.

**STEP #2.** Manually update the firewall or TCP-Wrapper for the illegal hosts or Ips.

**Problem:** Cannot be updated timely and may result in a compromise. Attacker may tries to intrude using some other service.

# IDEA

## Intelligent DEfence Activation

**WISH #1.** My server should be able to track attacker IP.

**WISH #2.** My server should be able to block attacker IP, automatically.

**WISH #3.** My server should secure itself, automatically, from any attack from attacker IP.

**Welcome BLOCKHOSTS!!**

# How it actually works

Someone do ssh with wrong password

|

Event is logged in /var/log/secure

|

blockhosts will check the log file for number of failed events

|

If number of failed events exceeds above limit

|

That hosts IP will be catch by blockhosts

|

Put that hosts entry in TCP-Wrapper and Iptables to block any further communication, for a time period.

# How to configure?

**STEP #1.** Download the latest version of denyhost from

**<http://www.aczoom.com/cms/blockhosts/>**

**STEP #2.** Install the package using RPM

**STEP #3.** Configure the main file **`/etc/blockhosts.cfg`**

# Configuring /etc/blockhosts.cfg

```
# python experts: script uses eval() to parse all the values specified here.
```

```
#-----
```

```
[common]
```

```
# common section is variables that may be used by main program, mail, etc
```

```
#HOSTS_BLOCKFILE = "/etc/hosts.allow"
```

```
# the name of the block-file on your computer - usually hosts.allow or
```

```
# hosts.deny, see "man 5 hosts_access" for details on these files.
```

```
# default is hosts.allow
```

```
#HOST_BLOCKLINE = ["ALL: ", " : deny"]
```

```
# the line to output, with Host Ip Address in between the strings above,
```

```
# to turn on blocking of that IP address. Must have 2 strings in list.
```

```
█
```

# Configuring /etc/blockhosts.cfg

```
[filters]
# filters section defines configuration for filtering watched hosts
# into the blocked hosts list

COUNT_THRESHOLD = 3
# number of invalid attempts after which host is blocked
# note that actual denial make take one or more attempts - depends on the
# timing of when LOGFILES are updated by the system, and when this script
# gets to run

AGE_THRESHOLD = 1
# number of hours after which host entry is discarded from hosts.deny
# 24 -> one day, 168 -> one week, 720 -> 30 days, integer values only
# most attackers go away after they are blocked, so to keep hosts.deny
# file size small, no reason to make this any more than, say, half-a-day

#WHITELIST = [ "127.0.0.1", ] # default
#WHITELIST = [ ]
#WHITELIST = [ "127.0.0.1", "172.24.0.252", "10\.0\.0\..*", ]
WHITELIST = [ "127.0.0.1", "172.24.0.252", ]
# A list of IP (IPv4) addresses or regular expressions that represent
# a IP (IPv4) address - this is the list of white-listed IP addresses.
# When considering IPs to block, if a IP address matches any item in this
# list, then it will be removed from the block list - so won't be blocked.
```

# Configuring /etc/blockhosts.cfg

```
#BLACKLIST = [] # default
#BLACKLIST = [ "192.168.10.1", "10\..*", ]
# A list of IP (IPv4) addresses or regular expressions that represent
# a IP (IPv4) address - this is the list of black-listed IP addresses.
# When considering IPs to block, if a IP address matches any item in this
# list, then it will be immediately added to the block list, even if
# COUNT_THRESHOLD may not have been reached.
# IP addresses directly specified in this list without a regular expression
# will be immediately added to the blocked list.
# WHITELIST takes precedence over BLACKLIST - so a match in both will mean
# it is white-listed.

#-----
[blockhosts]
# blockhosts section defines the log files to scan and patterns to look for

LOGFILES = [ "/var/log/secure", ] # default
#LOGFILES = [ "/var/log/auth.log", ]
#LOGFILES = [ "/var/log/secure", "/var/log/vsftpd.log", ]
# default list of logs to process, comma separated, can follow Python
# syntax, should be a sequence (list or tuple) of strings representing
# filenames: 1 or more files, default is single file: /var/log/secure
█
LOCKFILE = "/tmp/blockhosts.lock"
```

# Configure /etc/hosts.allow

```
#---- BlockHosts Additions
#---- BlockHosts Additions

sshd, proftpd, vsftpd: ALL: spawn /usr/bin/blockhosts.py \
  --echo "%c-%s" --ipblock=iptables \
  --whitelist="127.0.0.1" --blacklist="172.24.0.13"
```

# Start the blockhosts

```
[root@yourserver ~]# blockhosts.py --verbose
blockhosts 2.4.0 started: 2010-01-18 22:30:11 IST
... loaded /etc/hosts.allow, starting counts: blocked 0, watched 0
... loading log file /var/log/secure, offset: 4490
... discarding all host entries older than 2010-01-18 21:30:11 IST
... final counts: blocked 0, watched 0
[root@yourserver ~]# █
```

# Other similar tool



<http://denyhosts.sourceforge.net/>

# Remember:

All these tools works at “**Application Layer**” of the OSI layered model. So it need to work on 7 layers before taking a action.

**RIGHT!!!**

# Use **IPTables** to stop brute-force attacks - @ OSI layer - 4

IPTables can be used with **IPT\_RECENT** module to STOP brute-force attacks.

**STEP #1** – Download and install IPT\_RECENT module from:

**[http://www.snowman.net/projects/ipt\\_recent/](http://www.snowman.net/projects/ipt_recent/)**

# IPTables + IPT\_RECENT

## STEP #2 – Configure iptables

```
iptables -N SSH_CHECK
```

```
iptables -A INPUT -p tcp --dport 22 -m state --state NEW -j \  
SSH_CHECK
```

```
iptables -A SSH_CHECK -m state --state NEW -m recent \  
--set --name SSH
```

```
iptables -A SSH_CHECK -m state --state NEW -m recent \  
--update --seconds 60 --hitcount 4 --name SSH
```

```
iptables -A SSH_CHECK -m state --state NEW -m recent \  
--rcheck --seconds 60 --hitcount 4 --name SSH -j DROP
```

# IPTables + IPT\_RECENT

**STEP #3** – Save iptables configuration and reload “**ipt\_recent**” module.

```
service iptables save
```

```
modprobe ipt_recent
```