Advanced Linux Firewalls

Michael Rash
Security Architect
Enterasys Networks, Inc.

http://www.cipherdyne.org/

03/12/2008
SOURCE Boston, 2008
Agenda

• Intrusion Detection and Prevention via iptables
  – Snort rule emulation via iptables extensions (fwsnort)
  – iptables log analysis (psad)
• iptables log data visualizations
  – psad + AfterGlow + Gnuplot
• Single Packet Authorization + fwknop-1.9.2 release
• Live Demo
Why Talk about iptables in the Context of Intrusion Detection?

● Snort and commercial IDS infrastructure is mature (subject to usual concerns around false positives), but why stop there?

● IDS's can themselves be targeted, both from the detection and code execution standpoints
  - Modified Stick/Snot to send faked attacks over Tor
  - Snort DCE/RPC Preprocessor vulnerability

● Defense-in-depth is important

● Host fragment reassembly issues less of a concern for iptables string matching (more on this later)
IDS and iptables

• Can specify granular packet header tests, and logging format contains nearly all interesting packet header fields

• Can match against connection states
  - Useful for mitigating Stick/Snot style attacks

• String matching in the kernel started in the 2.4 days (patch applied via Netfilter patch-o-matic); made available again in 2.6.14
IDS and iptables (cont'd)

• Kernel textsearch (linux/lib/ts_*) infrastructure
  – Boyer-Moore and Knuth-Morris-Pratt algorithms

• String matching enabled by default in recent Linux kernels

• You get network layer defragmentation for free when connection tracking is used – you don't have to rely on proper configuration of frag3; it *is* the defragmentation algorithm of the host

• String matching within the filter table happens after network defrag
How About Intrusion Prevention?

- Plenty of reasons *NOT* to respond (false positives, possibility of attacker abuse, possibility of fingerprinting the response mechanism)

- However:
  - Can envision scenarios where controlling the shape of application layer data that can talk to local sockets is a good thing – iptables can enforce the DROP target (this is *prevention* instead of just some weak *response* mechanism)
  - Some automated attacks do not bother with obfuscation/encryption – target rich environment
  - Sometimes it is not easy to patch a production server whose uptime must remain high (assuming a patch even exists)
fwsnort

• Translates Snort signatures into “equivalent” iptables rules using string match extension and Netfilter connection tracking subsystem

• All translated Snort signatures are placed within user-defined chains, to which packets are jumped from built-in chains (INPUT, OUTPUT, and FORWARD)

• Maintains strict separation from existing iptables policy

• Approximately 60% of all Snort-2.3.3 rules (remember this is an IDS supplement) can be translated
fwsnort (cont'd)

- Emulation of Snort config variables such as $HOME_NET and $EXTERNAL_NET
- Reporting via LOG target (integrates with psad)
- Whitelists via the RETURN target
- Blacklists via the DROP or REJECT targets
- Snort signature info stored with the iptables comment match in kernel-space
- iptables is inline by definition; easy to configure fwsnort to use the DROP or REJECT targets
psad

- iptables log analyzer
- Email and syslog reporting
- Fwsnort integration
- DShield integration
- iptables LOG visualization with AfterGlow and Gnuplot
- Built-in passive OS fingerprinting derived from p0f (requires --log-tcp-options)
- IP options decoding (requires --log-ip-options)
psad (cont'd)

• Can detect Snort signatures that do not require application layer tests (source routing attempts, low ttl values, ICMP source quench, Nachi worm, etc.). This is all possible by virtue of iptables LOG format completeness.

• Detection of many port scan types generated by Nmap

• Timeout-based auto-blocking (optional, and can be restricted to application layer matches with fwsnort)

• Whitelists/Blacklists
iptables Packet Flow
fwsnort Packet Flow

Incoming Packets (jumped immediately to the FWSNORT_FORWARD chain from the FORWARD chain)

WHITELIST, non-ESTABLISHED, or completed fwsnort ruleset inspection

BLACKLIST packets as early as possible

DROP

Outgoing Packets

Copyright (C) 2008 Michael Rash
Example Snort Rule: nmap Execution via Web Server

alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS

(msg:"WEB-ATTACKS nmap command attempt";
flow:to_server,established; content:"nmap%20"; nocase;
classtype:web-application-attack; sid:1361; rev:5;)

Copyright (C) 2008 Michael Rash
fwsnort Translation

$iptables -A FWSNORT_FORWARD_ESTAB -d 192.168.10.0/24 -p tcp --dport 80 -m string --string "nmap%20" --algo bm -m comment --comment "msg: WEB-ATTACKS nmap command attempt; classtype: web-application-attack; rev: 5; FWS:0.9.0;" -j LOG --log-tcp-options --log-prefix "[1] SID1361 ESTAB "
“BLEEDING-EDGE VIRUS”
Signature (Multiple Content Fields)

alert tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any (msg: "BLEEDING-EDGE VIRUS Trojan-Spy.Win32.Bancos Download"; flow: established,from_server; content:"[AspackDie!]"); content:"|0f 6d 07 9e 6c 62 6c 68 00 d2 2f 63 6d 64 9d 11 af af 45 c7 72 ac 5f 3138 d0|"; classtype: trojan-activity; reference:url,securityresponse.symantec.com/avcenter/vennc/data/pwsteal.bancos.b.html; sid: 2001726; rev:6; )
(translated)

$IPTABLES -A FWSNORT_FORWARD_ESTAB -d 192.168.10.0/24 -p tcp --sport 80 -m string --string "[AspackDie!]" --algo bm -m string --hex-string "|0f 6d 07 9e 6c 62 6c 68 00 d2 2f 63 6d 64 9d 11 af af 45 c7 72 ac 5f 3138 d0|" --algo bm -m comment --comment "msg: BLEEDING-EDGE VIRUS Trojan-Spy.Win32.Bancos Download; classtype: trojan-activity; reference: url,securityresponse.symantec.com/avcenter/venc/data/pw steal.bancos.b.html; rev: 6; FWS:0.9.0;" -j LOG --log-ip-options --log-tcp-options --log-prefix "[640] SID2001726 ESTAB "
Supported Snort Rule Options

- All Snort rule header options
- content
- flow (conntrack)
- flags
- offset
- depth
- dsize (length match)
- itype
- icode
- ttl (ttl match)
- tos (tos match)
- ipopts
- ip_proto
- resp
Unsupported Snort Rule Options: Lost in Translation

- pcre
- flowbits
- byte_test <-- u32 module (coming soon – 2.6 support added)
- byte_jump <-- u32 module (coming soon – 2.6 support added)
- asn1
- window <-- included in iptables logs
- isdataat
- id <-- included in iptables logs
Unsupported Snort Rule Options (cont'd)

- icmp_id  <-- included in iptables logs
- icmp_seq  <-- included in iptables logs
- seq  <-- included with --log-tcp-sequence
- ack  <-- included with --log-tcp-sequence
- sameip  <-- included in iptables logs

- There are a few others - those that are logged can be analyzed by psad
Introducing iptables Logs

TCP
UDP
ICMP
iptables TCP Log Message

Mar 11 20:21:22 minastirith kernel: [199] SID1361 ESTAB IN=eth1 OUT=
MAC=00:13:d3:38:b6:e4:00:13:46:c2:60:44:08:00 SRC=192.168.10.3 DST=192.168.10.1 LEN=60
TOS=0x00 PREC=0x00 TTL=63 ID=11112 DF
PROTO=TCP SPT=28778 DPT=80 WINDOW=5840
RES=0x00 ACK PSH URGP=0 OPT
(0101080A02A041D20CC386B1)
iptables IP Header Coverage

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>IHL</td>
<td>Type of Service (TOS=, PREC=)</td>
<td>Total Length (LEN=)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Identification (ID=)</td>
<td>Flags (DF, MF)</td>
<td>Fragment Offset (FRAG=)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time To Live (TTL=)</td>
<td>Protocol (PROTO=)</td>
<td>Header Checksum</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source Address (SRC=)</td>
<td>Destination Address (DST=)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Options (OPT=, not decoded, requires -log-ip-options)</td>
<td>Padding</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# iptables TCP Header Coverage

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Port (SPT=)</td>
<td></td>
</tr>
<tr>
<td>Destination Port (DPT=)</td>
<td></td>
</tr>
<tr>
<td>Sequence Number (SEQ=)</td>
<td>requires --log-tcp-sequence</td>
</tr>
<tr>
<td>Acknowledgement Number</td>
<td>requires --log-tcp-sequence</td>
</tr>
<tr>
<td>Data Offset</td>
<td>Reserved (RES=), ECN (CWR,..), Flags (SYN, etc.)</td>
</tr>
<tr>
<td>Window (WINDOW=)</td>
<td></td>
</tr>
<tr>
<td>Checksum</td>
<td></td>
</tr>
<tr>
<td>Urgent Pointer (URGP=)</td>
<td></td>
</tr>
<tr>
<td>Options</td>
<td>requires --log-tcp-options</td>
</tr>
</tbody>
</table>

Copyright (C) 2008 Michael Rash
Passive OS Fingerprinting

• Required IP/TCP header fields for p0f:
  - Initial TTL
  - TCP window size
  - DF bit
  - SYN packet size
  - TCP options and order specification
p0f Signature Match with psad

Mar  8 23:23:48 minastirith kernel: DROP
IN=eth0  OUT= MAC=00:13:46:3a:41:4b:
00:90:1a:a0:1c:ec:08:00  SRC=208.53.138.16
DST=71.N.N.N  LEN=60  TOS=0x00  PREC=0x00  TTL=55
ID=23249  DF  PROTO=TCP  SPT=54155  DPT=3128
WINDOW=5840  RES=0x00  SYN  URGP=0  OPT
(020405B40402080A04C4FF5B00000000001030307)

S4:64:1:60:M*,S,T,N,W7:  Linux:2.6:8:Linux 2.6.8 and newer
iptables UDP Log Message

Mar 11 20:50:54 minastirith kernel: [153] SID2001597 IN=eth0 OUT=
MAC=00:13:d3:38:b6:e4:00:13:46:c2:60:44:08:00 SRC=192.168.10.3 DST=192.168.10.1 LEN=40
TOS=0x00 PREC=0x00 TTL=63 ID=29758 DF
PROTO=UDP SPT=32046 DPT=61 LEN=20
iptables UDP Header Coverage

<table>
<thead>
<tr>
<th>Source Port (SPT=)</th>
<th>Destination Port (DPT=)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length (LEN=)</td>
<td>Checksum</td>
</tr>
</tbody>
</table>
iptables ICMP Log Message

Mar 11 20:57:18 minastirith kernel: [98]
SID2003294 IN=eth0 OUT=
MAC=00:13:d3:38:b6:e4:00:13:46:c2:60:44:08:00 SRC=192.168.10.3 DST=192.168.10.1
LEN=128 TOS=0x00 PREC=0x00 TTL=63 ID=53466 PROTO=ICMP TYPE=8 CODE=0 ID=27459 SEQ=0
iptables ICMP Header Coverage

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 |
| Type (TYPE=)  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Code (CODE=)  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Checksum      |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| DATA: (depends on Type and Code and is variable length - logged to some extent) |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
How About an iptables Log Data Source?

Honeynet Project Scan Challenges
Honeynet Scan Challenge #34

- Challenge summary:
  - Challenge information and analysis can be found here: http://www.honeynet.org/scans/scan34/
  - Both Snort and iptables log data made available to the community (39MB of iptables data)
  - Contains port scans, port sweeps, traffic from worms, and outright compromises of Honeynet systems
Port Sweep Visualization

psad -m iptables.data --gnuplot --
CSV-fields src:not11.11.0.0/16
dp:count --gnuplot-graph points --
gnuplot-3d --gnuplot-view 74,77 --
gnuplot-file-prefix portsweep
Visualizing Port Sweeps (IP vs. Destination Port vs. Packet Count)
The Top Port Sweeper:
200.216.205.189 vs. TCP/3306
Honeynet Visualizations: Compromised Hosts

- Look for outbound connections from honeynet hosts with AfterGlow (see http://www.secviz.org)

```
# psad --CSV -m iptablelessyslog --CSV-fields "src:11.11.79.0/24 dst dp" | perl afterglow.pl -c color.properties |neato -Tgif -o outbound_connections.gif
```
Nachi Worm Visualization

• Look for 92-byte ICMP echo requests

# psad --CSV -m iptablelessyslog --CSV-fields "src dst ip_len:92" --CSV-max 300 --CSV-regex "PROTO=ICMP.*TYPE=8" | perl afterglow.pl -c color.properties |neato -Tgif -o nachi_worm.gif
Enhancing iptables Log Data

• Use --log-ip-options
• Use --log-tcp-sequence
• Use --log-tcp-options

- More attacks can be detected, and operating systems can be passively fingerprinted
Passive Authorization

• Basic idea:
  - Combine a default-drop packet filter with a passive mechanism to authenticate (and authorize) clients
  - The security benefit is derived from a reduction in the complexity of code that an arbitrary IP address can interact with. Every function has a non-zero probability of containing a security vulnerability
  - This is NOT security through obscurity; this is concealment (similar to passwords and encryption keys)
Port Knocking

• Uses packet headers to transmit information => serious protocol limitations
  • Difficult to protect against replay attacks
  • Low data transfer capability implies asymmetric encryption is not feasible
  • Knock sequences trivially busted from any source with spoofed duplicate packets
  • Port knocking sequences look like port scans to any IDS/IPS that is watching
Single Packet Authorization

• “Next-generation port knocking”
  - Uses application layer data
  - Replay attacks easily thwarted
  - Supports asymmetric ciphers
  - Only a single packet is transmitted, so much less likely to trigger IDS/IPS alarms
fwknop Features

• fwknopd “server” includes support for iptables and ipfw firewalls (Linux, Mac OS X, and FreeBSD).

• fwknop client includes support for Linux, Mac OS X, FreeBSD, Windows 2000, XP (under Cygwin) or via the Windows UI (developed by Sean Greven)

• SPA packets are encrypted either via Rijndael or with an asymmetric algorithm supported by GnuPG

• Supports outbound and inbound NAT (SNAT and DNAT, with DNAT support new in fwknop-1.9.0)
New in fwknop-1.9.2

- Client-derived firewall access timeouts
- Removal of encoded “Salted__” prefix from Rijndael SPA packets
- Support for Linux “cooked” interfaces (e.g. PPPoE)
- Selectable digest algorithms for replay attack detection (SHA256, SHA1, or MD5)
- Blacklist exclusions for SPA packets

- Special thanks to the SPAPICT team (Calsoft security enthusiasts + students from the Pune Institute of Computing Technology: http://tech.groups.yahoo.com/group/spapict/) who contributed many of the new features in fwknop-1.9.2.
fwknop Forward access via DNAT rules
fwknop SPA Packet Format

- random number (16 bytes)
- username
- timestamp
- software version
- message type and content:
  - 0 => command mode / command to execute
  - 1 => access mode / IP,proto,port
  - 2 => forward access mode / IP,proto,port / internalIP,externalNATPort
  - (optional) server_auth (post 0.9.2 release)
- message digest (SHA256 / SHA1 / MD5)
Example SPA Packets

- Clear text message (fields are base64 encoded before encrypted):
  - 5514438870168371:cm9vdA==:1203874973:1.9.2-pre6:1:MTI3LjAuMC4yLHRjcC8yMg==:yAynmuufyi/93SyVRViiB4MXKBhN/93cb+Ceu5cUuf4

- Two SPA packets (encrypted with the Rijndael cipher):
  - 9aoMEM9Jr5vHTdvKbx
    +phe3In6onbGLezoRpLD4y1YmcGW1udNGM1mAi/8b2s41aOhabyFvNzyxChfYSp7hPusjzLyRhwStmdZzFFazHxzNmBh9xsgAvrGLqmmQzYhS+
    +7XmtlH2D8hPjpaDGaGzs1nZPxFpZ2mQ5bjhBkutwcrkqCbe9wZf0o
  - /bucig8gNM4+WIDclkXkTyWJqEdEmHJwh
    +g4lrgAal09CYkPV9501z52zp00e/bRu5Oe/bKOJeD8hvEWK3LdOyVvuxfPWT9cDF7FG6xF/Rk4FhjcDPkaqVZb4CpMr7Yqr2wyL5Lxqy6YI7rt2ZdqaVGBIdGtzIHL
    OoXnz5j4mC1+H6hx7e0pO
Future Work

- Web proxy that creates SPA packets on behalf of anyone with a web browser
- Integration with the pf firewall on OpenBSD
- Integration with additional clients (scp, sftp, mail clients, etc.)
- Firefox SPA extension
  - fwknop is open source, please submit patches!
Live Demo...
References

- Raffael Marty's Blog and AfterGlow project: http://raffy.ch/blog/
- MRTG: http://oss.oetiker.ch/mrtg/ (psad support coming soon)
- Gnuplot: http://www.gnuplot.info/
References (cont'd)

• “An Analysis of Port Knocking and Single Packet Authorization”:
  http://www.securethoughts.net/spa/

• “Single Packet Authorization with fwknop”:
  http://www.cipherdyne.org/fwknop/docs/SPA.html

• “Enhancing Firewalls: Conveying User and Application Identification to Network
  Firewalls”: http://pages.cpsc.ucalgary.ca/~degraaf/

• Wikipedia on Port Knocking: http://en.wikipedia.org/wiki/Port_knocking

• Hakin9 on Port Knocking and SPA: http://mscoder.org/en/haking/articles_html.html

• Linux Journal articles:
  – http://www.linuxjournal.com/article/9565
  – http://www.linuxjournal.com/article/9621
Questions?

http://www.cipherdyne.org/

mbr@cipherdyne.org