An Achilles’ Heel in Signature-Based IDS: Squealing False Positives in SNORT

Sam Patton*  Bill Yurcik  David Doss
Department of Applied Computer Science
Illinois State University

*
Overview

1) the problem
2) software tool
3) generating FPs on Snort
4) implications
5) plausibility of attack
6) detection
7) the large issue of arbitrary F.P.generation
1) Problem: False Positives

- the vulnerability of IDS to false positives is well-documented
- we agree with Axelsson
  
  “the limitations of IDS is not the ability to accurately detect misuse behavior but rather the ability to suppress false alarms”
Testing IDS/Firewalls

- test correctness of IDS/firewall configuration
- simulate client-server characteristics of various attacks without using actual exploit code.
- better simulation of tcp connections to make attack simulations more stateful
- dangerously close to feature creep
  - # of new attacks increasing each month
  - false positive rates increasing proportionately
- may not be the right approach
2) Software Tool

- Purpose of tool
- Naming of the tool
  - Packet Creation Program
  - negative connotation
    - SATAN, crack
  - units are packets
  - packets are inhaled/snorted
  - user perception of extraordinary abilities

Phencyclidine

RAID 2001
PCP Overview

- Architecture
  - series of reusable C routines
  - series of generation tools

- Usage
  - recreate simulations of packets which might cause problems in IOS-based routers
  - testing of Firewall rule base configuration correctness
  - attack simulation
PCP Features

- Multi-protocol support within IP
- Standard support for tcp, udp, icmp
- Routing protocols
- Series of interesting sample scripts
- Currently working on better TCP sequence routine and RARP attacks
3) Generating FPs on Snort “Squealing”

- SNORT (1.6.3, 1.8.1) source code, user base
- Squealing (SNORT logo is a pig, noise made by pigs during times of distemperment)
  - The generation of network/host data with the intent of creating false positives in IDSs

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"IDS345 - OVERFLOW-NOOP-Sparc-TCP";flags:PA; content:"|13c0 1ca6 13c0 1ca6 13c0 1ca6 13c0 1ca6|";)
tcp_payload="\x13\xc0\x1c\xa6\x13\xc0\x1c\xa6\x13\xc0\x1c\xa6\x13\xc0\x1c\xa6\x13\xc0\x1c\xa6"

[**] IDS345 - OVERFLOW-NOOP-Sparc-TCP [**]
04/16-12:07:12.896655 10.0.3.254:23 -> 10.0.3.101:23
TCP TTL:64 TOS:0x18 ID:0
*****PA* Seq: 0x7FFFFFFF  Ack: 0x7FFFFFFF  Win: 0x4470
```
Linux and OpenBSD boxes running PCP
SNORT sensors running off OpenBSD 2.8
4) Implications

- noise-masked attacks  (peer review: diversion?)
- attack misdirection
- non-repudiation attack
- target conditioning attack
- statistical poisoning attack

- also from peer review:
  - evasion?, decoy?, deception?
5) Plausibility of Squealing “in the wild”

- Environment for increased usage exists
- Tool availability increasing....knowledge needed to exploit this style of attack is decreasing.
  - CASL (Newsham Ptacek), STICK, SNOT, Trichinosis, libnet, nemesis, idswakeup
- Tool maturity and evasiveness
6) Detection of “Squealing”

- detection today vs. detection tomorrow
  - static defaults (Nemesis) will be replaced with pseudo-random defaults
- increased reliance on RFC 2267 filters
  - noticing signatures egress from network
  - need for link-based ingress filtering for detection....not just reliance on router-based filters
- better simulation of two-way sessions from a single host
7) Future Protection From Arbitrary F.P. Generation

- adaption (randomness)
- state awareness
  - BUT as attack simulators become better at emulating state, the problem of squealing will reappear
- thinking about NIDS design model
  - the concept of data direction (timestamped)
2.0) A quick primer on writing false network data

- Two primary methods
  - SOCK_RAW and sendto()
  - bpf writes
2.1) Writing to the Network Layer

- 1) malloc() memory
- 2) Fill data structures and copy into buffer
- in the 3) Open a raw socket
  - fd = socket(AF_INET, SOCK_RAW, proto);
- 4) Call sendto()
  - c = sendto(fd, buffer, len, 0, (struct sockaddr *)&sin, sizeof(struct sockaddr));
- 5) Byte order issues on certain systems
  - i.e. Sometimes ip_length and ip_offset in host byte order
2.2) Writing to the Data Link

- Get Datalink type
  - ioctl(l->fd, BIOCGDLT, (caddr_t)&v)

- Linkoffset
  - l->linkoffset = 0xe;  /*ethernet */

- Open file descriptor
  - l->fd = open(device, O_RDWR);

- Attach to bpf device
  - (ioctl(l->fd, BIOCSETIF, (caddr_t)&ifr)

- Writing to link layer
  - c = write(l->fd, buffer, len);
5.1) Detection of Squeals Today

<table>
<thead>
<tr>
<th><code>enet_src[0]</code></th>
<th><code>enet_dst[0]</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02</td>
<td>0x0d</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0f</td>
<td>0x0e</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0a</td>
<td>0x0a</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0d</td>
<td>0x0d</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0e</td>
<td>0x00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0d</td>
<td>0x01</td>
</tr>
</tbody>
</table>
5.2) Detection of Squeals

Today

\[ \text{fl\_opt} = 0; \]
\[ \text{sport} = 42069; \]
\[ \text{dport} = 23; \]
\[ \text{id} = 0; \]
\[ \text{urgp} = 2048; \]
\[ \text{tos} = \text{IPTOS\_LOWDELAY} \mid \text{IPTOS\_THROUGHPUT}; \]
5.3) Detection of Squeals

Today

\[
\begin{align*}
ttl & = 254; \\
seq & = 420; \\
ack & = 420; \\
win & = 512; \\
payload & = \text{NULL}; \\
*options & = \text{NULL}; \\
frag & = \text{IP\_DF};
\end{align*}
\]