Intrusion detection

- Firewalls provide security around the perimeter of networks
  - Control traffic going in and out of a local network
- Traditional firewalls = packet filters
  - Analyze packet headers & enforce policy
  - Reject packets that violate policy
- But malware can still get in
  - Application-layer attacks
  - Misconfiguration
  - Internal deployment via web downloads, attachments, USB drives
- Intrusion Detection System (IDS) / Intrusion Prevention System (IPS)
  - Monitor entire packets: header and payload, searching for known events
  - IDS: log & alert
  - IPS: log & alert but also reject packets

Modes of detection

- Anomaly-based detection
  - Know normal behavior
  - Unusual activity is bad
- Misuse detection
  - Know bad behavior
  - Anything else is good

Anomaly-based detection

- Monitor network or system activity
- Classify it as "normal" or "anomalous" (possibly bad)
- Detection based on rules or heuristics
  - System needs to be told — or learn — what is normal
  - Sometimes AI techniques can be used to build statistical baselines
- May generate false positives
  - You download files from a new website in a "suspicious" area

Misuse-based detection

- Also monitor network or system activity
- Bad activity patterns are embedded in rules called signatures
  - Yet another use of the word
  - Signature = encryption with a private key
  - Signature = portion of virus code to be matched
  - Signature = patterns of activity
- Detection is accurate
  - ... but cannot detect unknown attacks

Capturing packets

If you want to monitor all traffic on the local network via a host:
- Ethernet switches route traffic directly to the destination port
- You need to:
  - Configure your switch port for monitor mode to receive all traffic
  - Configure your Ethernet transceiver to promiscuous mode to relay traffic to the OS
Snort

One of the most widely used network intrusion detection systems

- Free & open source
- Uses packet sniffing (examining network traffic)
  - Via Unix's libpcap or Windows' WinPcap libraries for packet sniffing
- Uses rules to combine signature & protocol inspection methods
  - Some anomaly detection – as long as it can be codified: no learning

Three components

1. Packet Decoder
   - Extracts data from raw network traffic
   - Selects items that may be of interest and can be used for rule construction

2. Detection Engine
   - Rule set is applied to each captured packet
   - Rules organized in a linked list: headers & options

3. Logging or Alerting
   - Extracted packet data

Snort Rules

- Header contains:
  - Action: tells Snort what to do when it finds a packet that matches the criteria
    - alert: generate an alert using a selected alert method & log the packet
    - log: log the packet
    - activate: alert and then turn on a dynamic rule
    - drop: block and log the packet
    - reject: block the packet, log it, and send a TCP reset or ICMP "unreachable"
    - sdrop: drop the packet but do not log it
  - Protocol, source address, destination address, source port, destination port
  - Options (e.g., patterns, TTL, payload size)

- Activate & dynamic rules
  - Record activities that occur after a certain event takes place
    - Activate rule: activates a second rule
    - Dynamic rule: starts collecting & logging packets

Rules format

Simple but flexible rule definitions: fixed headers and zero or more options

<table>
<thead>
<tr>
<th>Header</th>
<th>Option fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>IP TTL</td>
</tr>
<tr>
<td>Protocol</td>
<td>Content offset</td>
</tr>
<tr>
<td>IP addresses</td>
<td>Content depth</td>
</tr>
<tr>
<td>TCP/UDP ports</td>
<td>PORE (Perl-Compatible regular expressions)</td>
</tr>
<tr>
<td>Traffic direction</td>
<td>Session recording</td>
</tr>
<tr>
<td></td>
<td>TCP seq number</td>
</tr>
<tr>
<td></td>
<td>TCP ack number</td>
</tr>
<tr>
<td></td>
<td>Payload size</td>
</tr>
<tr>
<td></td>
<td>Content</td>
</tr>
<tr>
<td></td>
<td>Alternate log files</td>
</tr>
</tbody>
</table>

Options

- Options are processed using logical AND
  - ... all conditions in a rule must apply

- Content offset & depth can be set to limit the amount of data to search

- Content (byte values) & PCRE (Perl-style regular expressions) matching options take the most time and are performed last

PHF = Sample CGI program included with Apache
Sample Rule

```
alert tcp !192.168.1.0/24 any -
>
>              192.168.1.0/24 111 (content:"|00 01 86 a5|"; msg:"external mountd access");
```

Match
- any IP address except anything from 192.168.1.0/24
- on any port
- with a destination of 192.168.1.0/24
- port 111
- using TCP

Sample Rule: notify of root ftp logins

```
alert tcp any any -
>
> any 21 (flow:to_server,established; content:"root"; pcre:"/users+root/.*";)
```

Match
- any source address and port
- any destination address
- port 21 (FTP port)
- using TCP
- Flow: traffic going to the server on an established TCP connection
- Content contains root – the most unique string in the attack
- Enables fast pattern matching – no need to test regular expression if root is missing
- Content contains "user", at least one space, followed by "root", ignoring case

3. Logging/Alerting

- Choice of formats for logging
  - Human readable format
  - tcpdump format

- Alerting
  - Send to syslog
  - Write to alert text file

- Logging/alerting can be turned off based on performance/annoyance needs

Where to get rules

- Without IDS rules, snort is just a packet sniffer
- You can write your own rules
- Snort.org has 23499 community rules for various known exploits
- Plus
  - Sourcefire-certified (now Cisco) rules
  - Bleeding Snort Rules (bleeding edge – beta – rules)
  - Other places … but watch out!
- Ruleset size continues to grow
  - Snort spends up to 80% of its time pattern matching

Anomaly Detection

- Anomaly detection via inference is difficult
- Not enough training data
  - We have a lot of data for normal activities
  - Not much for realistic attacks
- Even normal data drifts
  - Changes in behavior over time & legitimate unpredictable behavior
  - Attacker can attack incrementally
- Normal activities not well understood
  - Attack may be in the bounds of normal statistics
- False alerts are costly
  - System administrators will spend a lot of time poring over data

The end