A Software Tool for Network Intrusion Detection

4th Biennial Conference

Presented by: Christiaan van der Walt

Date: October 2012
Presentation Outline

- Need for intrusion detection systems
- Overview of attacks
- Illustration of network traffic for various attacks
- Simulation of data
- Description of NetID Algorithm
- Illustration of NetID Software Tool
- Future Work
The Need for Network Intrusion Detection Systems

- Online services and security of data
- Serve content -> serve applications
- Online services include internet banking, e-commerce, video streaming, Gmail
- Data services include Dropbox, Google Docs, Google Drive
- Threats: hacking, Denial of Service (DoS) attacks
- Victims of DoS attacks include Yahoo, eBay, e-trade, CNN
- Distributed DoS attacks
- Why another software tool?
Types of DoS attacks

• Consumption of computational resources – bandwidth, disk space, processor time

• Disruption of configuration information – routing information

• Disruption of state information – unsolicited resetting of TCP connections

• Obstructing communication media – users and victim can’t communicate adequately
Commonly used attacks

- TCP SYN (Neptune) flooding attack
  - More than 90% of DoS attacks use the TCP protocol
  - SYN flood is the most commonly-used TCP attack
  - Exploits the limitation of the three-way hand shake, that maintains half-open connections for a certain time period
  - Neptune - SYN flood denial of service on one or more ports
Commonly used attacks

- **ICMP (Smurf) attack**
  - Use spoofed broadcast ICMP echo (ping) messages
  - Sends spoofed ping messages (spoof ip address so that it seems the victim of the attack is sending them)
  - The network to which these ping requests are sent, then forwards these requests to all hosts in the network and each of them respond with an echo reply, thus multiplying the traffic by the number of hosts

- **Teardrop attack**
  - Exploits the flaw in the implementation of TCP/IP stacks that cannot handle overlapping IP fragments
  - Sends mangled IP packets with overlapping IP fragments, and large payloads to victim of attack
What do DoS Attacks Look Like?

Figure: Network packet rate for different attacks
Approaches to detecting DoS flooding attacks

• Adaptive threshold algorithms
  • Monitor the traffic flow (number of packets per second), in case of SYN flood, they monitor the SYN packet rate
  • When packet rate exceeds a threshold a possible intrusion is flagged
  • Threshold is adapted to account for daily and weekly variations, typically make use of mean packet rate of recent traffic

• Change-point detection algorithms
  • Based on hypothesis testing for iid data
  • Continually estimate a statistical distribution of network traffic, and test whether the change in distribution is statistically significant
Algorithm Development

- Make use DARPA Intrusion Detection Evaluation Data generated by MIT
- The simulated the following DoS attacks

<table>
<thead>
<tr>
<th>Name</th>
<th>Service</th>
<th>Vulnerable Platforms</th>
<th>Mechanism</th>
<th>Time to Implement</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache2</td>
<td>http</td>
<td>Any Apache</td>
<td>Abuse</td>
<td>Short</td>
<td>Crash httpd</td>
</tr>
<tr>
<td>Back</td>
<td>http</td>
<td>Any Apache</td>
<td>Abuse/Bug</td>
<td>Short</td>
<td>Slow server response</td>
</tr>
<tr>
<td>Land</td>
<td>N/A</td>
<td>SunOS</td>
<td>Bug</td>
<td>Short</td>
<td>Freeze machine</td>
</tr>
<tr>
<td>Mailbomb</td>
<td>smtp</td>
<td>All</td>
<td>Abuse</td>
<td>Short</td>
<td>Annoyance</td>
</tr>
<tr>
<td>SYN Flood</td>
<td>Any TCP</td>
<td>All</td>
<td>Abuse</td>
<td>Short</td>
<td>Deny service on one or more ports for minutes</td>
</tr>
<tr>
<td>Ping of Death</td>
<td>icmp</td>
<td>None</td>
<td>Bug</td>
<td>Short</td>
<td>None</td>
</tr>
<tr>
<td>Process Table</td>
<td>Any TCP</td>
<td>All</td>
<td>Abuse</td>
<td>Moderate</td>
<td>Deny new processes</td>
</tr>
<tr>
<td>Smurf</td>
<td>icmp</td>
<td>All</td>
<td>Abuse</td>
<td>Moderate/Long</td>
<td>Network Slowdown</td>
</tr>
<tr>
<td>Syslogd</td>
<td>syslog</td>
<td>Solaris</td>
<td>Bug</td>
<td>Short</td>
<td>Kill Syslogd</td>
</tr>
<tr>
<td>Teardrop</td>
<td>N/A</td>
<td>Linux</td>
<td>Bug</td>
<td>Short</td>
<td>Reboot machine</td>
</tr>
<tr>
<td>Udpstorm</td>
<td>echo/ chargen</td>
<td>All</td>
<td>Abuse</td>
<td>Short</td>
<td>Network Slowdown</td>
</tr>
</tbody>
</table>
Data Simulation Setup

```
Target machines

"Inside"
172.16.* eyrie.af.mil

"Outside"
192.168.1.* world.net
194.27.251.* plum.net
197.218.177.* grape.mil
```
NetID Online Detection Algorithm

- Detect attacks in real-time (is capable of scanning 4 hours of data in a few minutes)
- High detection accuracy and fast detection time (< 5 seconds)
- Performs change detection via an advanced statistics adaptive threshold algorithm

Algorithm

- Specify sampling rate (typically 5ms)
- Arrival times of network packets are converted to packet rate (packets per sampling rate)
- Specify window length (typically 80s)
- Estimate the probability density function (PDF) of the data in the time window starting at time $t$
- Specify a window step size (typically 2.5s)
- Move the samples in the window to time $t + \text{step}_\text{size}$
- Estimate the PDF of window $t + \text{step}_\text{size}$
- Calculate the change in distribution
- Test if change $> \text{threshold}$
Simulation results – Smurf Attack 1

[Graphs showing packet traffic and network traffic over time]
Simulation results – Neptune Attack 1
Simulation results – Smurf Attack 2

![Graphs showing network traffic and Smurf/Neptune attack over time]

- Network traffic
- Smurf/Neptune attack

![Graph showing intrusion metric over time]

- Intrusion metric
- Fixed threshold

![Graph showing traffic distribution over packets per second]

- Traffic distribution
- Packets per second (log scale)
NetID Software Tool (Illustration)
Operator Controls

Figure: Operator Tools

Figure: Parameter settings
Future Work

• Add more intrusion detection algorithms to NetID Software

• Investigate more attacks and their detection performance

• Simulate our own attacks (possibly from within NetID Software)

• Further develop the analysis tools for NetID
Thank you