Denial of Service Attacks

Notes derived from Michael R. Grimaila's originals
Denial Of Service

• The goal of a denial of service attack is to deny legitimate users access to a particular resource.

• An incident is considered an attack if a malicious user intentionally disrupts service to a computer or network resource.

• Resource exhaustion
Resource Exhaustion

- Disk Space
- CPU Cycles
- Memory
- Network Bandwidth
- Application Resources
  - TCP Stack
  - Web Connections
What’s the Harm?

• Financial loss can be difficult to estimate
  - Lost business
  - Bad publicity and damaged reputation
• 2002 CSI/FBI Survey
  - 40% of reported attacks are DOS
  - Average cost per attack is >$1 million
• Distributed DOS attacks (February 2000)
  - Amazon, CNN, E-Trade, eBay, etc...
  - Estimated losses were “several millions to billions of dollars”
• DOS can also be used to cover-up “real” attacks
Microsoft Web Sites Attacked
Company Asks FBI For Assistance

By Ariana Bunjing Cha and David Stratfield
Washington Post Staff Writers
Friday, January 26, 2001, Page B01

A hacker brought down Microsoft Corp.'s Web sites yesterday afternoon, only hours after the company fixed a technician's error that had disabled its sites for a full day.

Microsoft issued a statement decrying "this kind of illegal activity" and said it had called on the FBI for assistance. Almost a year ago, attackers used a similar strategy, called a "distributed denial of service," to temporarily paralyze such leading Web sites as Yahoo, CNN, eBay and Amazon.
'Mafiaboy' hacker jailed

A Canadian hacker nicknamed Mafiaboy has been sentenced to eight months in a youth detention centre.

Judge Giles Cuellet ruled that the 17-year-old teenager from Montreal committed a criminal act when he crippled major internet sites like Amazon and Yahoo last year, causing an estimated $1.7 billion in damages.

"The motivation was undeniable," the judge said. "The adolescent had a criminal intent."

The custodial sentence was welcomed by the prosecution as a strong message against the worldwide hacking community.
Arrested Goner Creators Left Obvious Online Trail

By Brian McWilliams, Newsbytes
SAN DIEGO, CALIFORNIA, U.S.A.,
09 Dec 2001, 1:15 PM CST

Despite efforts to hide behind nicknames and stolen Internet accounts, four Israeli teenagers responsible for creating the Goner Internet worm were easy to identify, investigators said.

The names of the teens, who were arrested Friday by police in the northern city of Nahariya, were not disclosed under Israeli law.

The four have confessed to writing the worm, which by one estimate has already caused $5 million damage since it hit the Internet Dec. 4.

Clues provided in Goner’s software code enabled investigators to track down the worm’s author and three accomplices in just over two days.

The creator of Goner published his online nickname, "Suid," along with three of the two individuals who tested the worm, "ThE_SKuLL" and "Isatan," as well as the nickname of Simon, a former victim.
Types of attacks

- There are three general categories of attacks.
  - Against users
  - Against hosts
  - Against networks
Local DOS against hosts

- fork() bomb
- intentionally generate errors to fill logs, consuming disk space, crashing
- The power switch!!
Local DOS: Countermeasures

- partition disks
- disk quotas
- set process limits
- monitor system activity/CPU/Disk Usage
- Physical Security
Network Based Denial of Service Attacks

- Most involve either resource exhaustion or corruption of the operating system runtime environment.
- UDP bombing
- tcp SYN flooding
- ping of death
- smurf attack
UDP bombing

- Two UDP services: echo (which echos back any character received) and chargen (which generates character) were used in the past for network testing and are enabled by default on most systems.

- These services can be used to launch a DOS by connecting the chargen to echo ports on the same or another machine and generating large amounts of network traffic.
**UDP service denial:**

**Countermeasures**

- Disable echo, chargen and all other unused services whenever possible, such as `/etc/inetd.conf` on Unix, and "no udp small-services" on Cisco IOS.

- Filter UDP traffic at the firewall level. Only allow legitimate traffic such as UDP port 53 (DNS)
Windows UDP attacks

- NewTear, Newtear2, Bonk, and Boink are tools that exploit the same weakness in the Microsoft Windows 9.x/NT TCP/IP stack.
- The attacker sends the victim a pair of malformed IP fragments which get re-assembled into an invalid UDP datagram. Upon receiving the invalid datagram, the victim host “blue-screens” and freezes or reboots (The pathologic offset attack)
- Countermeasure: Apply vendor patches
TCP SYN Flooding

• Also referred to as the TCP “half-open” attack
• To establish a legitimate TCP connection:
  - the client sends a SYN packet to the server
  - the server sends a SYN-ACK back to the client
  - the client sends an ACK back to the server to complete the three-way handshake and establish the connection
TCP SYN Flooding (cont’d)

- The attack occurs by the attacker initiating a TCP connection to the server with a SYN. (using a legitimate or spoofed source address)
- The server replies with a SYN-ACK
- The client then doesn’t send back a ACK, causing the server to allocate memory for the pending connection and wait.
  (If the client spoofed the initial source address, it will never receive the SYN-ACK)
TCP SYN Flooding: Results

- The half-open connections buffer on the victim server will eventually fill.
- The system will be unable to accept any new incoming connections until the buffer is emptied out.
- There is a timeout associated with a pending connection, so the half-open connections will eventually expire.
- The attacking system can continue sending connection requesting new connections faster than the victim system can expire the pending connections.
TCP Three-Way Handshake

Client connecting to a TCP port

Client initiates request

SYN
Client wishes to establish connection

SYN-ACK
Server agrees to connection request

ACK
Client finishes handshake

Connection is now half-open

Server connection Established

Client connection Established
SYN Flood Illustrated

Client spoof request

Client SYN Flood

S
半开

S
半开

S
半开

S
半开

S
队列已满

S
队列已满

S
队列已满

S
I have ACKed these connections, but I have not received an ACK back!
SYN Flood Protection

• Cisco routers
  - TCP Intercept
    • Intercepts SYN packets and proxies to server
    • Knits two half-connections together if successful

• Checkpoint Firewall-1
  - SYN Defender
    • Similar to Cisco’s TCP Intercept

• Determined attacker might still succeed
  - Exhaust resources on router or firewall
TCP Intercept Illustrated

Request connection

Finishes handshake

Server answers

Knit half connections
SYN Flood Prevention

• Increase the listen queue
  - Implementation depends on OS

• Aggressive timeouts
  - Expire half-open connections sooner
  - Might impact clients on congested networks

• Use an OS impervious to this attack

• Apply all vendor patches
SYN Flood remedies

• Use a cache of half-open connections
• When the cache is full, drop waiting half-opens randomly
  – Send a RST to the sender
  – Legitimate users reinitiate the connection
• Impact of floods reduced
  – Service still denied probabilistically
SYN Cookies

• The server doesn’t maintain half-open state
• Sends the client a sequence number in the ACK
  – ISN carries most of the information of the initial SYN request
• If Client completes the 3-way handshake
  – Will return the server’s ISN
  – Server will use this to complete the connection
More on SYN Cookies

• All the data about connection need to be encoded into the ISN – 32bits

• TCP requires some properties on ISN
  – Should be random
  – Should be dependent on time
  – Not easily guessable

• How to reconcile these?
Linux SYN cookie algorithm

\[ H_1 = \text{hash}_{32-61}(S_{addr}|S_{port}|D_{addr}|D_{port}|K_1) \]
\[ H_2 = \text{hash}_{32-61}(S_{addr}|S_{port}|D_{addr}|D_{port}|\text{counter}|K_2) \]

**Generation:**
\[ \text{cookie} = H_1 + ISN_{client} + (\text{counter} \times 2^{24}) + (H_2 + \text{data}) \mod 2^{24} \]

**Validation:**
\[ \text{counter}_{\text{cookie}} = (\text{cookie} - H_1 - ISN_{client}) \div 2^{24} \]
\[ \Delta\text{counter} = \text{counter}_{\text{current}} - \text{counter}_{\text{cookie}} \]
\[ \text{data} = (\text{cookie} - H_1 - ISN_{client}) \mod 2^{24} - H_2 \mod 2^{24} \]

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hash_n (x)</td>
<td>n bit range, starting from lsb0, produced from x using the compression function of a digest algorithm (MD5 or SHA-1)</td>
</tr>
<tr>
<td>S_addr, S_port</td>
<td>source TCP/IP address</td>
</tr>
<tr>
<td>D_addr, D_port</td>
<td>destination TCP/IP address</td>
</tr>
<tr>
<td>K_1, K_2</td>
<td>secret keys</td>
</tr>
<tr>
<td>ISN_client</td>
<td>ISN provided by the client in the SYN segment</td>
</tr>
<tr>
<td>counter</td>
<td>minute counter</td>
</tr>
<tr>
<td>data</td>
<td>24-bit value</td>
</tr>
</tbody>
</table>
Linux SYN cookie algorithm

- $K_1$ – 52byte key, $K_2$ – 48byte key
- Data value is 0,1,…7
- Two checks: $\text{counter should be } +ve, < \text{ allowed time}$
- Data value should be between 0,1,…,7

<table>
<thead>
<tr>
<th>SYN Cookie data</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSS value</td>
<td>64</td>
<td>256</td>
<td>512</td>
<td>536</td>
<td>1024</td>
<td>1440</td>
<td>1460</td>
<td>4312</td>
</tr>
</tbody>
</table>
SYN Cookies

• Occasionally possible to have a collision
• Not easy to support TCP options
• Read more here:

Improving the functionality of SYN cookies
by Andre Zuquate
Ping of Death

- The TCP/IP specification allows for a maximum packet size of 65,536 octets.
- The ping of death attack sends oversized ICMP datagrams (encapsulated in IP packets) to the victim.
- Some systems, upon receiving the oversized packet, will crash, freeze, or reboot, resulting in denial of service.
- Countermeasures: Most systems are now immune, but apply vendor patches if needed.
When Smurfs go bad!!

- A smurf attack consists of a host sending an ICMP echo request (ping) to a network broadcast address. (usually network addresses with the host portion of the address having all 1s)

- Every host on the network receives the ICMP echo request and sends back an ICMP echo response inundating the initiator with network traffic.
Smurf Attacks

• There are 3 players in the smurf attack
  - the attacker, the intermediary (which can also be a victim) and the victim
• In most scenarios the attacker spoofs the IP source address as the IP of the intended victim to the intermediary network broadcast address.
• Every host on the intermediary network replies, flooding the victim and the intermediary network with network traffic.
• Result: Performance may be degraded such that the victim, the victim and intermediary networks become congested and unusable
Smurf Attack Example

Honey! I think our network is having another Smurf attack!
Smurf Example

1. Attacker sends ICMP packet with spoofed source IP
   Victim → 10.1.2.255

2. Attacker sends ICMP packet with spoofed source IP
   Victim → 192.168.1.255

3. Victim is flooded with ICMP echo responses

4. Victim hangs?
Smurf: Countermeasures

• Configure routers to deny IP broadcast traffic onto your network from other networks. In almost all cases, IP-directed broadcast functionality is not needed.
• Configure hosts (via kernel variable) to NOT reply to a packet sent to a broadcast address
• Configure Ingress/Egress filters on routers to counteract IP address spoofing.
Distributed Denial of Service Attacks (DDOS)

- Attacker logs into Master and signals slaves to launch an attack on a specific target address (victim).

- Slaves then respond by initiating TCP, UDP, ICMP or Smurf attack on victim.
Consequences of UDP floods

Unfairness

- When UDP and TCP compete, UDP wins by pushing TCP into congestion
Unfairness - FIFO

X-axis: UDP Arrival Rate (% of R1-R2). Dashed Line: UDP Arrivals; Dotted Line: UDP Goodput; Solid Line: TCP Goodput; Bold line: Aggregate Goodput
Unfairness - WRR

Figure 3: Simulations with three TCP flows and one UDP flow, with WRR scheduling. There is no unfairness.
Loss of goodput - FIFO

- Packets dropped later in network

Figure 6: Simulations with one TCP flow and three UDP flows, showing congestion collapse with FIFO scheduling.
Loss of goodput - WRR

Figure 7: Simulations with one TCP flow and three UDP flows, showing congestion collapse with WRR scheduling.
Why are UDP floods bad?

- Hard to separate legitimate UDP traffic (multimedia, DNS) from DOS traffic
- Easy to generate
  - A PC can easily saturate a 1Gbps link
- Network stability at risk
Why are DOS attacks possible?

• IP employs an open architecture
  - No authentication of sender’s IP address
  - Easy to forge any address, hard to detect offender
  - IP traceback, ingress/egress filters (later)

• No resource regulation in the network
  - Employ QoS techniques to mitigate impact (later)
Distributed Denial of Service Attacks (DDoS)

- trin00 (WinTrinoo)
- Tribe Flood Netowrk (TFN) (TFN2k)
- Shaft
- stacheldraht
- Mstream
Trin00

- Affects Windows and many Unix OS's
- Attacker scans for exploits, gains root, and downloads Trin00 programs.
- Attacker -> Master -> Daemon hierarchy (One -> More -> Many)
- Attacker can telnet into a Master to initiate commands, which are distributed amongst its Daemons.
Trin00 (con’t)

- Communication between Master->Daemon through a password-protected cleartext UDP-based protocol.
- Daemons attack the target with a UDP or TCP packet bombardment.
- Used in the February 2000 attacks on eBay, Amazon, CNN, etc.
TFN (2k)

- Smurf attack
- ICMP flood
- SYN flood
- UDP flood
- All three at once
Stackeldraht

- ICMP flood
- SYN flood
- UDP flood
- Smurf attack
Shaft

- ICMP flood
- SYN flood
- UDP flood
- All three at once
DOS Toolkits

• Distributed DOS programs
  - Trinoo, Tribe Flood Network (TFN), Stacheldraht, Shaft
  - Many agents attack a single target

• Source code archives
  - packetstorm.securify.com
  - Rootshell.org
  - Newsgroups

• Do a Google search for Denial of Service

• Look at this talk
DDOS: Countermeasures

- RID:
  - Sends out packets and listens for reply
  - Detects Trinoo, TFN, Stacheldraht

- NIPC - find_ddos tool
  - Runs on local system
  - Detects Trinoo, TFN, TFN2k

- Bindview’s Zombie Zapper
  - Tells DDOS slave to stop flooding traffic
Conclusions

• Denial of Service attacks are one of the most difficult attacks to defend against
• Damages can be significant for eCommerce and eServices
• Can be used as a diversion to confuse incident response teams
• Our nations critical infrastructure is also at risk
• Tool kits are readily available on the Internet... Check it out using Google!