# Foundations of Network and Computer Security

**J**ohn Black

Lecture #13
Oct 7th 2004

CSCI 6268/TLEN 5831, Fall 2004

# Announcements

- Two new readings were assigned
  - How to 0wn the Internet in Your Spare Time
  - Thompson’s Turing award lecture
    - Both on our web site

- If you still need your Quiz #2, I have it

- Project #1 due Oct 19
  - Use OpenSSL **`verify`** command to verify CA signature on your cert
    - We didn’t discuss this, but you can look it up

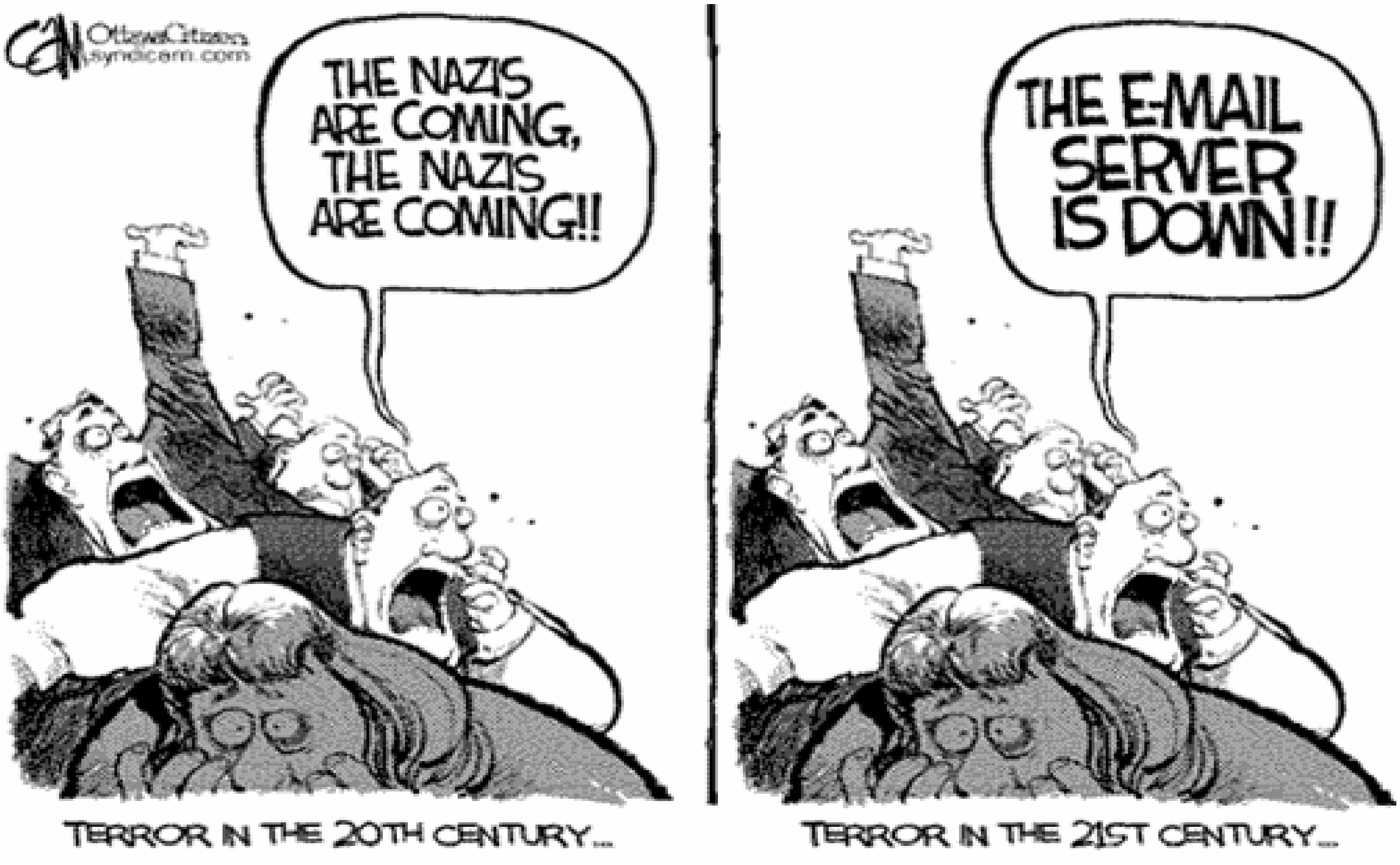
- Midterm a week from today
  - Material: lectures through today; all readings; all projects (not silly OpenSSL details)

# Denial of Service

- An old idea
  - Picket lines, blockades, doorbell ditch, false pizza orders, prank phone calls, etc.

- First technological DoS I know of
  - Denver Taxi company in the 50’s
  - Promised a white driver every time
  - Civil rights protesters called and left phone off hook
    - Tied up phone lines back then

# Modern Reliance on Computers


CAM Ottawa Citizen syndicam.com
THE NAZIS ARE COMING, THE NAZIS ARE COMING!!
TERROR IN THE 20TH CENTURY...
THE E-MAIL SERVER IS DOWN!!
TERROR IN THE 21ST CENTURY...

# DoS (cont)

- In the computer arena
  - Mail bombs
    - Large emails to fill up someone’s hard disk
  - Network traffic
    - Lots of bogus traffic aimed at just overwhelming victim
    - This is typically not TCP traffic
      - Why not?

# Network-Based DoS

- Common methods
  - Large UDP packets
    - Max size is 65,536 bytes
    - Will fragment over IP and all frags hit victim
    - Victim tries to reassemble IP fragments
  - ICMP echo
    - Aka “ping”
    - Can also be large
    - (“Ping of death”)

# SYN Floods

- A TCP-based method
  - Normal TCP handshake starts with SYN from client
  - Causes server to make an entry in the “SYN queue” and use up some time
  - SYNs are very small, so attacker sends a ton of them
  - A SYN at the server is called a “half-open connection”
    - These eventually time out, but it takes a while

# First Attempted Remedy: Filtering

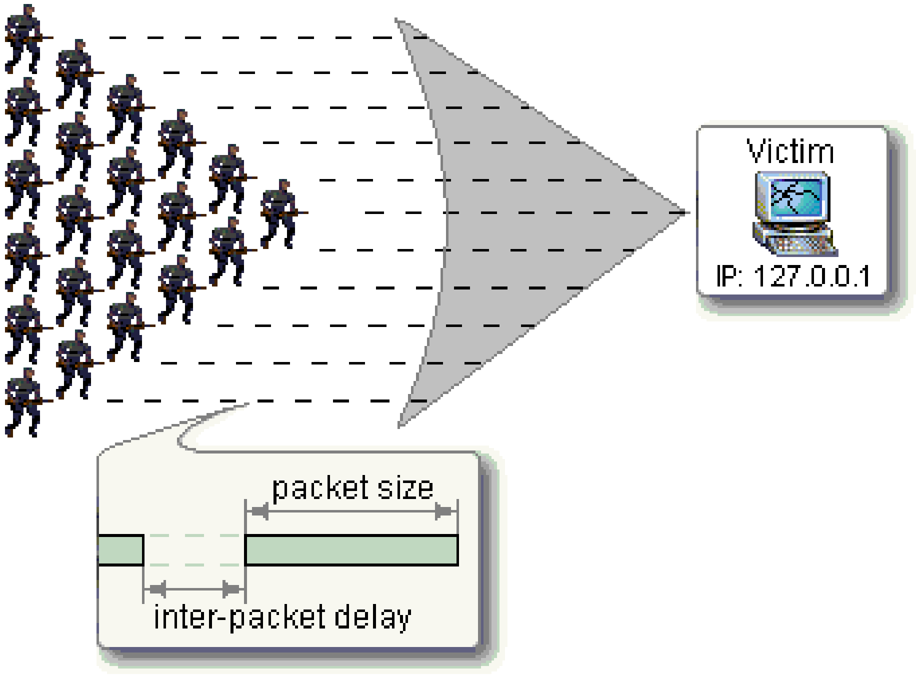
- Victim can try and filter out the IP source address of the attacker
  - This has to be done upstream or the victim’s connection bandwidth is saturated
    - If ISP is willing to install a filter on the appropriate source address, this works
  - But attacker can spoof source IP
    - Attacker is not completing any TCP association, and *wants* to leave connections half-open
    - This is almost always done

# Reflection Attacks (aka “Smurfing”)

- Technique for amplifying traffic
  - Often works behind firewalls as well
  - Instead of flooding victim V with SYNs, we send SYNs to hosts $H_1$, $H_2$, …, $H_n$ and spoof the source address as V
    - (Here n is large… say, 1000 or more)
    - Hosts send SYN/ACK to V
    - V is very confused and reacts in various ways
    - If hosts are behind firewall, it appears as though attack is coming from local machines
    - Hosts are usually not overwhelmed, so they don’t feel the attack

# DDoS: Distributed DoS

- Now, multiple attackers



# DDoS

- Most famous attack was in Feb 2000 against Amazon, Yahoo, eBay, and other major e-commerce sites
- Estimated losses of $1.2 billion US
- Easy for almost anyone to launch
  - Most of these, by the way, are hackers attacking other hackers

# Recruiting “Zombies”

- A “Zombie” is a computer which has been captured by the attacker
  - Typically by a virus or by just using some vulnerability
- Each infiltrated computer receives a hidden program from the “Zombie Master”
- The Zombie Master keeps a list of which computers he has control over
- When the time comes, he instructs all of his Zombies to simultaneously attack the victim computer

# Case Study: The Gibson Story

- Who is Steve Gibson?
  - Owns Gibson Research Corp (grc)
  - Old time programmer
  - Self-proclaimed security expert
  - Writes tools in assembly (!)
  - Has taken on Microsoft for raw sockets in XP
    - More on this later
  - Some don’t like him (www.grcsucks.com)

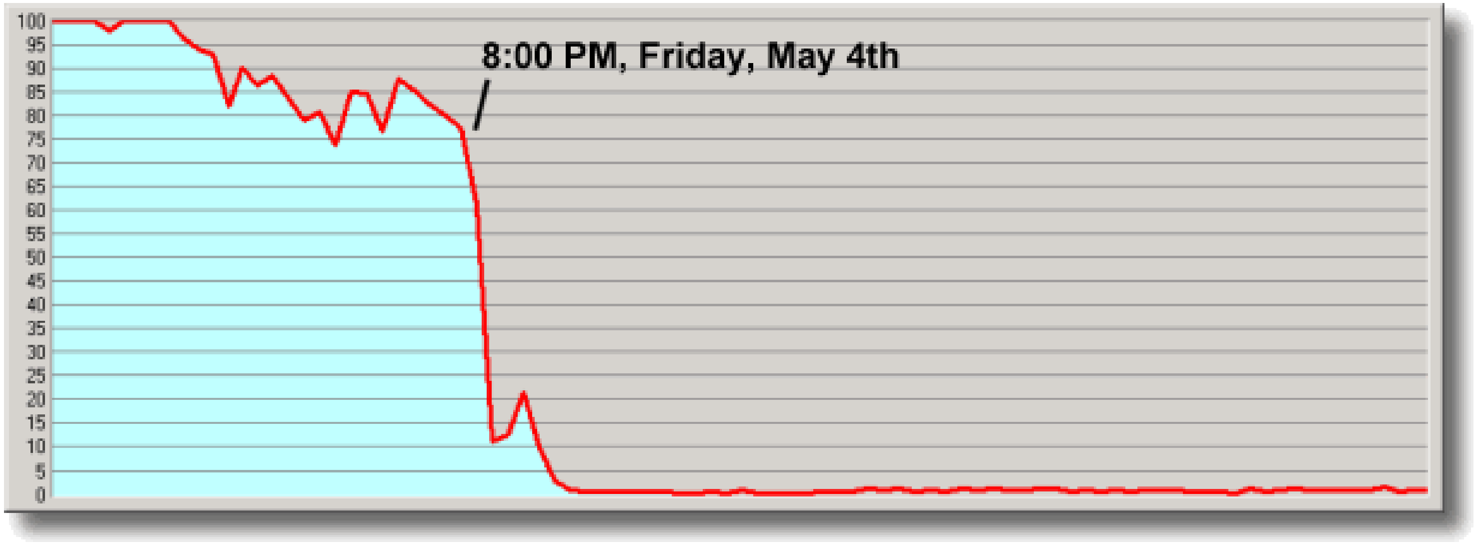
# The GRC Story

DENIAL OF SERVICE

- Please read this article; it’s on our web page.
- It’s kind of wordy, but fun and informative reading.

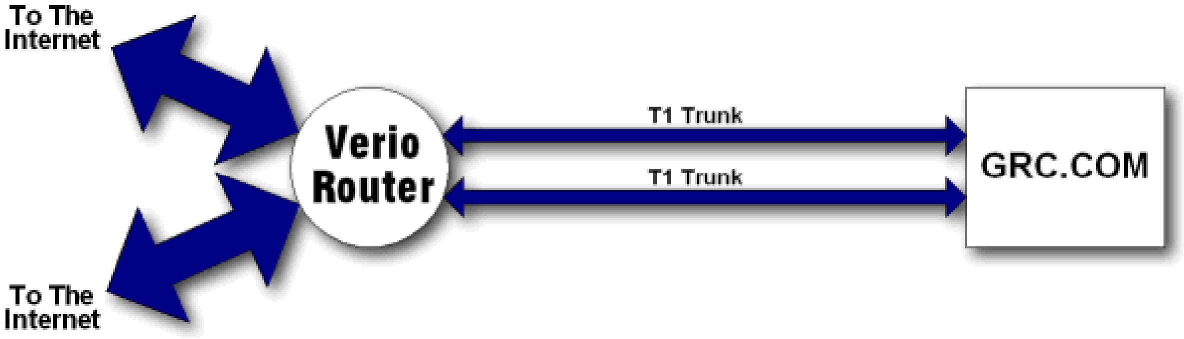
# The Story

- At 8pm on Friday May 4th, 2001, grc.com disappeared from the Internet



# DDoS Attack

- T1 trunks are 1.54Mbit/sec
- Verio has 100Mbit/sec connections to Internet
- UDP traffic aimed at port 666
  - Large packets which had fragmented into 1500-byte chunks
  - Firewall discarded it, but still saturated T1's
  - Need to filter at Verio's end



# Filtering

- After some calls, filtering is in place
  - Verio blocks UDP and ICMP
    - Until Win2K and XP, it was difficult to send SYNs
      - Sending SYN’s would have been hard to block since this would have meant shutting down all TCP connections, including http to grc.com
    - Raw sockets in Win2K and XP mean that spoofing source IP is now possible, which also makes it harder to filter (as mentioned already)

# Zombies

- Gibson tracked 474 source addresses sending the packets
  - All compromised windows machines
  - Most popular are cable-modem boxes
    - Always on, high bandwidth

| | |
|---|---|
| 104 | home.com |
| 51 | rr.com |
| 20 | aol.com |
| 20 | mediaone.net |
| 17 | uu.net |
| 14 | btinternet.com |
| 14 | shawcable.net |
| 14 | optonline.net |
| 14 | ne.jp |
| 9 | chello.nl |
| 9 | ntl.com |
| 8 | videotron.ca |
| 7 | ad.jp |
| 7 | psi.net |
| 6 | uk.com |

# Attacks Continue

- Attacker re-targets
  - First goes to IP of firewall
    - This is different from the IP of the grc.com server
    - Verio shuts down ICMP and UDP to this IP as well
  - Then goes to the Cisco router
    - Since it's STILL on the grc side of the T1, it again knocks grc.com off the network
  - How is the attacker getting these IP addresses?
    - Traceroute

# Size of the Attack

- Verio filtered 2.4 billion fragmented UDP datagrams headed for port 666
  - grc.com was completely unaware of the attack
  - Filtering relied on being able to track source IP addresses
    - Would not have worked if attacker had spoofed source IP, which is commonly done nowadays

# Meet the Attacker

- The attacker, it turns out, was 13
  - “Wicked” said he was attacking grc.com because Gibson had allegedly referred to a friend as a “script kiddie”
  - Gibson said he did not and asked the attacks to stop
    - They did for a while, but resumed
  - Gibson tracked the IP to a town in Wisconsin, but needed Earthlink to cooperate to find the actual phone number
    - They wouldn’t

# @Home

- Gibson tried to contact @home
  - Wanted to inform them of the 100+ users who were compromised
    - And from which he was being attacked
  - No useful response
    - They wanted the IPs, but wouldn’t let Gibson have the zombie code, which he really wanted
    - Gibson refused to give up the IPs

# The Feds

- Gibson calls the FBI
  - Until $5,000 of damage had been done, no crime had even been committed.
  - Secondly, they said that even if they did manage to meet the $5,000 minimum required for "Wicked's" activities to qualify as criminal, their staffs were overloaded and swamped with cases involving companies that had lost huge sums of money to Internet crime.
  - Finally, they said that since "Wicked" was only 13 years old, nothing much would happen to him, even if the preponderance of evidence demonstrated that he was behind the attacks. They said that a couple of agents might go out to his home and have a talk with his parents, but in this country his youth was an impenetrable shield.

# The Feds (cont)

- Gibson gave the IPs to the Feds

- Gibson also noted that Wicked was in trouble at age 8, had computer removed until age 10, was monitored for 1 year and was now back at it

# Gibson gets the Bot

- Bot was anonymously dropped into a mailbox
  - It was an IRC client
    - Password-protected chat room
    - 100’s of other machines came and went as Gibson waited
  - Suddenly Gibson’s test machine was attacking Finland
  - Gibson decides to “neuter” the zombie and just monitor it

# Long Story Short

- Eventually he meets more senior hackers
  - Surprised to see him in the chat room
  - Gets them to talk to Wicked
  - Attacks stop
- He learns a lot about the Sub7Server trojan
  - Invasive zombie program
    - Monitors key strokes for password, cc#s, etc.
    - **netstat -an | find ":6667"**

```
TCP 192.168.1.101:1026 70.13.215.89:6667 ESTABLISHED
```

# How Common are DDoS Attacks?

- Backscatter Analysis
  - http://www.caida.org/outreach/papers/2001/BackScatter/index.xml
    - I'm not assigning this as reading
  - Idea is that almost all spoofed traffic uses a randomly generated source IP
    - All popular DDoS attack tools do this
      - trinoo, TFN, TFN2k, Stacheldraht, etc.
  - When replies from victim are sent, they go to this (bogus) source IP

# Backscatter Technique

- CAIDA (San Diego) owns large block of IP address space
  - They have a lightly-used Class A network
  - They assumed
    - All source addresses uniformly chosen
      - Misses reflection attacks
    - All attack packets reliably reach victim
    - All replies reliable leave victim
    - Any unsolicited replies seen by CAIDA were backscatter

# Approach

- Backscatter packets revealed
  - Type of attack
    - SYN/ACK means SYN flood
    - ICMP messages from routers indicated other types of attacks like UDP floods
  - IP of victim
    - Source address of backscatter
  - Intensity of attack
  - Duration of attack

# Results

- 12,805 distinct attacks against over 5,000 hosts in 2,000 organizations in three weeks
- About 6000 packets per sec on average

# DDoS: Preventative Measures

- Tracing and filtering
  - If source addresses could not be forged, filtering would be a reasonable solution
- Ingress filtering
  - Idea: if you are an ISP, don’t let packets leave your IP address space if they have source addresses out side your address space
  - Old idea
  - Simple
  - Still a lot of ISPs don’t do this
  - Even with ingress filtering, attackers can jump around within a range of IP addresses
  - Note that this limitation meant some backscatter numbers were probably a bit off

# SYN Cookies

- A SYN flood leaves half-open connections
  - The “SYN queue” is a data structure which keeps track of these half-open connections
  - We track the source IP and port of client, server IP and port, seq# of client, seq# of server
  - Idea: we don’t really need to keep all of this
    - We just need enough to recognize the ACK of the client
    - Can we get away without storing *anything* locally?

# SYN Cookies: The Idea

- Store nothing locally
  - ISN: Initial sequence number
  - Encode all we need to remember in the ISN we send back to the client
  - t: a 32-bit counter which increments every 64 seconds
  - K: a secret key selected by server for uptime of server
  - Limitations: MSS limited to 8 values

*Server ISN*

| | | |
|---|---|---|
| ***t mod 32*** | ***MSS*** | ***hash(client IP and port \|\| server IP and port \|\| t \|\| K)*** |
| *5* | *3* | *24* |

# SYN Cookies: Details

- MSS: Maximum Segment Size
  - Suggested by client, server then computes best value
    - Details depend on whether they are on the same network, MTU on network, etc
    - Server can have only 8 values to encode here
- What happens when client replies with ACK?
  - Client will reply with ISN+1 of server in the ACK
  - Server then subtracts 1 and checks against hash of client IP and port, server IP and port, t which matches in the lowest 5 bits, and K
    - If match, put in SYN queue
    - If not, ignore

# SYN Cookies: Limitations

- Note that this will NOT prevent bandwidth-saturation attacks
  - This technique seeks only to prevent SYN queue overflows
  - If attacker can saturate bandwidth, this doesn’t help
    - But note that bandwidth saturation is not going to be TCP-based
    - UDP and ICMP can be used for bandwidth saturation, but we are often less dependent on these protocols

# SYN Cookies: Implementation

- Standard in Linux and FreeBSD

```
echo 1 > proc/sys/net/ipv4/tcp_syncookies
```

- As far as I know, Windows does not implement them yet

# Tracebacks Methods

- One basic problem with fighting DDoS is that we cannot find the *source IP* of the attacker
  - Finding the attacker would allow us to shut down the attack at the source
    - This assumes ISPs will cooperate and that there is a mechanism in place for reporting the source
    - Both of these assumptions are questionable as we saw in the Gibson story
  - The Internet Protocol (IP) makes it hard to find out where things are coming from
    - Easy to forge source IPs
    - No tracing mechanism available
      - This is on purpose

# Adding Traceback

- Perhaps we could add a mechanism to IP to implement traceback
  - Needs to be backward-compatible with current routing protocols
    - If not, too expensive and no one will do it
  - There have been several suggestions
    - Probabilistic traceback
    - Algebraic traceback
    - Others
  - We'll look just at probabilistic traceback

# Probabilistic Traceback