**Topics Covered in Course**

**Attacks and hacker tools**
- Reconnaissance
- Network mapping
- Port scanning
- Sniffing
- IP address spoofing
- Session hijacking
- DoS

**Defenses**
- Intrusion detection
- Cryptography
- Firewalls
- Secure protocols
- PGP
- SSL
- IPsec (and VPN)
- WEP

*Before talking about defenses, need to look at network from attacker’s perspective*

---

**Reconnaissance**

- "casing the joint"
- A few words about low-tech reconnaissance:
  - Social engineering: duping an employee into revealing sensitive info.
  - Physical break-in and trash searches
  - Getting info from Web and Usenet

*Let’s take a close look at:*
- Reconnaissance with whois
- Reconnaissance with DNS

---

**Reconnaissance: Whois databases**

- Registrar: organization at which you register a domain name
  - Verifies uniqueness of name
  - Enters domain name into various databases: whois & DNS
  - Internet Corporation for Assigned Names and Numbers (ICANN) accredits registrars
    - See www.internic.net

---

**List of registrars from internic.net:**
Whois databases

- Input: domain name or company name
- Output: registrar, whois server, dns server

Some useful whois sites:
- www.internic.net
  - For com, net and org top-level domains
- www.allwhois.com
  - For country-code top-level domains, e.g., jp, fr

Two steps
- First find target’s registrar
- Then whois target at registrar

Internic Whois: Target “kazaa”

Whois: next step

Do whois at registrar, e.g., www.register.com

- Input: domain name, IP address, net administrator name
- Output:
  - Names of people (administrator, billing contact)
  - Telephone numbers
  - E-mail addresses
  - Name servers and IP addresses

Whois at kazaa’s registrar
Reconnaissance: IP Ranges

- ARIN: American Registry for Internet Numbers
  - Maintains WHOIS database that includes IP address ranges in US
- RIPE: Europe
- APNIC: Asia

Why WHOIS databases need to be publicly available

- If you’re under attack, can analyze source address of packets (if source address is not spoofed).
- Can use WHOIS database to obtain info about the domain from where the attack is coming.
- Can inform admin that their systems are being used to launch an attack

Reconnaissance: DNS database

Let’s quickly review DNS:

- Distributed database implemented in hierarchy of many DNS servers
- Application-layer protocol: host, DNS servers communicate to resolve names (address/name translation)
  
Authoritative name server:

- For a given domain (e.g., poly.edu), provides server name to IP address mappings for servers in domain
- Primary and secondary name server for reliability
Introduction

Polytechnic University

DNS: queries

Requesting host

cis.poly.edu

dns.cs.umass.edu

galia.cs.umass.edu

Root DNS Servers

com DNS servers

org DNS servers

edu DNS servers

yahoo.com DNS servers

amazon.com DNS servers

pbs.org DNS servers

dns.poly.edu

local DNS server

dns.poly.edu

DNS protocol, messages

Name, type fields for a query

RRs in response to query

Records for authoritative servers

Additional "helpful" info that may be used

Query and reply messages sent

Over UDP on port 53

DNS records

DNS: distributed db storing resource records (RR)

RR format: (name, value, type, ttl)

- **Type=A**
  - name is hostname
  - value is IP address

- **Type=MX**
  - name is name of mailserver associated with name

- **Type=NS**
  - name is domain (e.g. foo.com)
  - value is IP address of authoritative name server for this domain

Polytechnic University
**DNS: caching and updating records**
- Once (any) DNS server learns mapping, it *caches* mapping.
  - Cache entries timeout (disappear) after some time.

**nslookup**
- Available in most Unix & Windows machines.
- Get dialpad DNS server IP address from whois.
- Set type=any "get all".

**Interrogating DNS servers**
- Attacker first gets primary or secondary authoritative server for target organization using whois.
- Attacker can then query the DNS by sending DNS query messages.
- Tools (often available in Unix and Windows machines; also available at web sites):
  - nslookup
  - host
  - dig

**Defenses from DNS reconnaissance**
- DNS is needed to map server names: dns, Web, e-mail.
- Additional, unnecessary information is not required:
  - Host name should not indicate OS type.
- Restrict zone transfers:
  - Configure BIND properly.
  - Firewalls should block TCPs Port 53 (used for zone transfers).
- Use "split DNS":
  - Some systems need to be public.
  - Other systems are internal.
**Introduction**

**Polytechnic University**

---

**Split DNS**

When internal host needs to make external query, internal DNS becomes a proxy.

---

**Sam Spade: General Purpose Reconnaissance**

**Tool: www.samspade.org/ssw**

- ping
- whois
- nslookup
- dig
- dns zone transfer
- traceroute
- finger
- SMTP VRFY

---

**Reconnaissance summary**

- Obtaining information from public databases:
  - whois databases
    - Tool: web sites
  - DNS database
    - Tool: nslookup
- Defense
  - Keep to a minimum what you put in the public database: only what is necessary

---

**Topics Covered in Course**

**Attacks and hacker tools**

- Reconnaissance
- Network mapping
- Port scanning
- Sniffing
- IP address spoofing
- Session hijacking
- DoS

**Defenses**

- Intrusion detection
- Cryptography
- Firewalls
- Secure protocols
  - PGP
  - SSL
  - IPsec (and VPN)
  - WEP

Before talking about defenses, need to look at network from attacker's perspective.
Sweeping: Finding Live Hosts

- Used whois to determine IP address range
- Used nslookup to find some servers
- Let's now try to find the other live hosts

Ping Sweep

Ping
- Recall ICMP messages are directly encapsulated in IP datagrams (protocol 1)
- To ping a host:
  - send ICMP Echo Request (ICMP type 8)
  - Host responds with ICMP Echo Reply (type 0)
- So let's ping the entire IP address range
  - Use automated tool for this ping sweep
- If firewall blocks ping packets:
  - Try sweeping with TCP SYN packets to port 80
  - Or try sending UDP packets to possible ports

Traceroute

 traceroute: gaia.cs.umass.edu to www.eurecom.fr

1 cs-gw (128.119.240.254)  1 ms  1 ms  2 ms
2 borderl-rt-fa5-1-0.gw.umass.edu (128.119.3.145)  1 ms  1 ms  2 ms
3 cti-vbns.gw.umass.edu (128.119.3.130)  6 ms  5 ms  5 ms
4 j1-atl-0-0-19.wor.vbns.net (204.147.132.129)  16 ms 11 ms 13 ms
5 j1-so7-0-0.wae.vbns.net (204.147.138.136)  21 ms 18 ms 18 ms
6 abilene-vbns.abilene.ucaid.edu (198.32.11.0)  22 ms 18 ms 22 ms
7 nycm-wash.abilene.ucaid.edu (198.32.8.46)  22 ms 22 ms 22 ms
8 65.40.103.253 (65.40.103.253)  104 ms 109 ms 106 ms
9 de2-1.de1.de.geant.net (62.40.96.129)  109 ms 102 ms 104 ms
10 de.fr1.fr.geant.net (62.40.96.50)  113 ms 114 ms 114 ms
11 renater-gw-gw1.fr.geant.net (62.40.103.54)  112 ms 114 ms 112 ms
12 qro-n2.cssi.renater.fr (193.51.206.13)  111 ms 114 ms 116 ms
13 nice.cssi.renater.fr (195.220.98.102)  123 ms 125 ms 124 ms
14 r3t2-nice.cssi.renater.fr (195.220.98.115) 126 ms 126 ms 124 ms
15 eurecom-valbonne.r3t2.ft.net (193.48.50.54) 135 ms 128 ms 133 ms
16 194.214.211.25 (194.214.211.25) 126 ms 128 ms 126 ms
17 **
18 * * *
19 fantasia.eurecom.fr (193.55.113.142) 132 ms 128 ms 136 ms

Traceroute: How it works

- Source sends UDP packets to target
  - Each to an unlikely port
  - 3 packets with the same TTL, then increments TTL
- When router decrements TTL to 0, sends back to source ICMP packet
  - type 11, code 0, TTL expired
- When target receives packet, sends back to source ICMP packet
  - type 3, code 0, destination port unreachable
Learning about network topology

- Attacker often uses traceroute to determine path to each host discovered during ping sweep.
- Overlay results from traceroute to create an approximate network diagram.

Cheops: ping, traceroute, picture

Defenses against network mapping

- Filter using firewalls and packet-filtering capabilities of routers:
  - Block incoming ICMP packets, except to the hosts that you want to be pingable.
  - Filter Time Exceeded ICMP messages leaving your network.

Topics Covered in Course

- **Attacks and hacker tools**
  - Reconnaissance
  - Network mapping
  - Port scanning
  - Sniffing
  - IP address spoofing
  - Session hijacking
  - DoS

- **Defenses**
  - Intrusion detection
  - Cryptography
  - Firewalls
  - Secure protocols
  - PGP
  - SSL
  - IPsec (and VPN)
  - WEP
**Port scanning**

- Now that we have a map with some hosts, let's find out what ports are open on a target host.
- 65,535 TCP ports; 65,535 UDP ports:
  - Web server: TCP port 80
  - DNS server: UDP port 53
  - Mail server: TCP port 25
- Port scanning tools can scan:
  - List of ports
  - Range of ports
  - All possible TCP and UDP ports
- Attacker may scan a limited set of ports, to avoid detection.

---

**Interlude TCP segment structure**

<table>
<thead>
<tr>
<th>Source port</th>
<th>Dest port</th>
<th>Sequence number</th>
<th>Acknowledgement number</th>
<th>Flags</th>
<th>Options</th>
<th>Receive window</th>
<th>Checksum</th>
<th>Urgent Pointer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Interlude: TCP seq. #’s and ACKs**

**Seq. #’s:**
- byte stream "number" of first byte in segment’s data

**ACKs:**
- seq # of next byte expected from other side

**Simple telnet scenario**

<table>
<thead>
<tr>
<th>User types</th>
<th>Host A</th>
<th>Host B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Seq=42, ACK=79, data = ‘C’</td>
<td>Seq=79, ACK=43, data = ‘C’</td>
</tr>
<tr>
<td></td>
<td>host ACKs receipt of echo ‘C’</td>
<td>host ACKs receipt of echo ‘C’</td>
</tr>
</tbody>
</table>

---

**Interlude: TCP Connection Establishment**

**Three way handshake:**

1. **Step 1:** Client host sends TCP SYN segment to server
   - SYN=1, ACK=0
   - specifies initial seq #
   - no data
2. **Step 2:** Server host receives SYN, replies with SYN-ACK segment
   - SYN=1, ACK=1
   - server host allocates buffers
   - specifies server initial seq. #
3. **Step 3:** Client receives SYN-ACK, replies with ACK segment, which may contain data
   - SYN=0, ACK=1
**TCP: Reset packet**
- If machine receives a TCP packet it is not expecting, it responds with TCP packet with RST bit set.
  - For example when no process is listening on destination port
  - For UDP, machine returns ICMP “port unreachable” instead

**Nmap: Port scanning tool (1)**
- [www.insecure.org/Nmap]
  - Polite scan: TCP connect
  - Attempts to complete 3-way handshake with each target port
  - Sends SYN, waits for SYNACK, sends ACK, then sends FIN to close connection
  - If target port is closed, no SYNACK returned
    - Attacker typically receives RST packet
  - TCP connect scans are easy to detect
    - Target (e.g. Web server) may log completed connections
    - Gives away attacker’s IP address

**Nmap: Port scanning tool (2)**
- Stealthier: TCP SYN Scans
  - Send SYN, receive SYNACK, send RST
    - Send RST segment to avoid an accidental DoS attack
  - Stealthier: hosts do not record connection
    - But routers with logging enabled will record the SYN packet
  - Faster: don’t need to send FIN packet

**Nmap: Port scanning tool (3)**
- TCP ACK Scans
  - Many filters (in firewalls and routers) only let internal systems hosts initiate TCP connections
    - Drop packets for which ACK=0 (ie SYN packet): no sessions initiated externally
  - To learn what ports are open through firewall, try an ACK scan (segments with ACK=1)
Nmap: Port scanning tool (4)

- **UDP Scan**
  - UDP doesn't have SYN, ACK, RST packets
  - Attacker simply sends UDP packet to target port
    - ICMP Port Unreachable: interpret port closed
    - Nothing comes back: interpret port open
    - False positives common
    - For example, if NMAP indicates the UDP port 7070 is open, attacker may use RealPlayer client to enter

Nmap: Port scanning tool (5)

Obscure source of scan

- Attacker can enter list of decoy source IP addresses
- For each packet it sends, Nmap also sends packets from decoy source IP addresses
  - If 4 decoy sources, send five packets
- Note that attacker's actual address must appear in at least one packet, to get a result
- If there are 30 decoys, victim network will have to investigate 31 different sources

Nmap: Port scanning tool (6)

TCP Stack Fingerprinting (1)

- In addition to determining open ports, attacker wants to know OS on targeted machine:
  - exploit machine's known vulnerabilities
  - Sophisticated hacker may set up lab environment similar to target network
- TCP implementations in different OSes respond differently to illegal combinations of TCP flag bits.

TCP stack fingerprinting (2)

- Nmap sends
  - SYN to open port
  - NULL to open port (no flag bits set)
  - SYN/FIN/URG/PSH to open port
  - SYN to closed port
  - ACK to closed port
  - FIN/PSH/URG to closed port
  - UDP to closed port
- Nmap includes a database of OS fingerprints for hundreds of platforms
Defenses against port scanning

- Close all unused ports
- Scan your own systems to verify that unneeded ports are closed
- Firewalls (to be discussed later)

Topics Covered in Course

Attacks and hacker tools
- Reconnaissance
- Network mapping
- Port scanning
- Sniffing
- IP address spoofing
- Session hijacking
- DoS

Defenses
- Intrusion detection
- Cryptography
- Firewalls
- Secure protocols
- PGP
- SSL
- IPsec (and VPN)
- WEP

Before talking about defenses, need to look at network from attacker's perspective

Sniffing

- Attacker is inside firewall
- Requirements
  - Attacker's host connected to shared medium
  - NIC should be in "promiscuous mode" processes all frames that come to NIC
- Sniffer has two components
  - Capture
  - Packet analysis

- Grab and file away:
  - userids and passwords
  - credit card numbers
  - secret e-mail conversations
- Island hopping attack:
  - Take over single machine (eg buffer overflow)
  - Install sniffer, observe passwords, take over more machines, install sniffers

Passive sniffing

- Easy to sniff:
  - 802.11 traffic
  - Ethernet traffic passing through a hub
    - Any packets sent to hub is broadcast to all interfaces
    - Not true for a switch
  - Cable modem traffic
- Popular sniffers
  - Ethereal (saw this in CS 684)
  - tcpdump (for unix)
  - Snort (sniffing and intrusion detection)
  - minisniff (used in lab)
Active sniffing through a switch

- **Switch**: Link layer device
  - stores and forwards Ethernet frames
  - examines frame header and selectively forwards frame based on MAC dest address
- **A switch has a switch table**: entry in switch table:
  - (MAC Address, Interface, Time Stamp)
  - stale entries in table dropped (TTL can be 60 min)
- **Switch learns** which hosts can be reached through which interfaces
  - when frame received, switch "learns" location of sender: incoming LAN segment
  - records sender/location pair in switch table

Switch review: example

Suppose C sends frame to D and D replies back with frame to C.

- **Switch receives frame from C**
  - notes in switch table that C is on interface 1
  - because D is not in table, switch sends frame into interfaces 2 and 3
- **Frame received by D**

Switch review: dedicated access

- **Switch with many interfaces**
- **Hosts have direct connection to switch**
Sniffing through a switch

How does attacker sniff packets sent to/from the victim?

Have to get victim's packets to attacker!

Sniffing through a switch: corrupting switch memory approach

Host sends flood of frames with random MAC addresses
- Switches memory gets filled with bogus MAC addresses
- When "good packet arrives," dest MAC address not in switch memory
- Switch broadcasts real packets to all links
- Sniff all the broadcast packets

Sniffing through LAN: poison victim's ARP table approach

(1) Send fake ARP response, mapping router IP address to attacker's MAC address
(2) Victim sends traffic destined to outside world. Poisoned ARP table causes traffic to be sent to attacker
(3) Packets are forwarded from attacker's host to default router
(0) Sniff all frames that arrive. Configure so that IP packets arriving from victim are forwarded to default router

Sniffing defenses

- Encrypt data: IPSec, SSL, PGP, SSH
- Get rid of hubs: complete migration to switched network
- Use encryption for wireless and cable channels
- Configure switches with MAC addresses
  - Turn off self learning
  - Eliminates flooding problem
- Honeypot
  - Create fake account and send password over network
  - Identify attacker when it uses the password
Defense to sniffing: Improper response to Ping

- Send ping (ICMP Echo request) to suspected sniffer with
  - IP address of sniffer
  - But incorrect MAC address
- Non-promiscuous node would drop frame
- But sniffer forwards payload to ICMP, which sends echo reply. Oops!
- Can also try with TCP SYN, which might generate SYNACK or RST
- Can design clever sniffers to counter
  - e.g., don’t return anything unless MAC address is correct

Topics Covered in Course

Attacks and hacker tools
- Reconnaissance
- Network mapping
- Port scanning
- Sniffing
- IP address spoofing
- Session hijacking
- DoS

Defenses
- Intrusion detection
- Cryptography
- Firewalls
- Secure protocols
  - PGP
  - SSL
  - IPsec (and VPN)
  - WEP

Before talking about defenses, need to look at network from attacker’s perspective

IP address spoofing (1)

- Attacker doesn’t want actions traced back
- Simply re-configure IP address in Windows or Unix.
- Or enter spoofed address in an application
  - e.g., decoy packets with Nmap

IP address spoofing (2)

- But attacker cannot interact with victim.
- Unless attacker is on path between victim and spoofed address.
- Question: If attacker sends SYN with spoofed IP address, what happens?
**Anti-Spoof Filters**

- Border routers should drop packets leaving network if source address is not part of network = "egress anti-spoof filtering"
  - Someone inside might be launching an attack.
  - Or someone outside... if attacker takes over one of your servers (eg, Web or DNS), attacker can send spoofed packets to another organization.

**Topics Covered in Course**

**Attacks and hacker tools**
- Reconnaissance
- Network mapping
- Port scanning
- Sniffing
- IP address spoofing
- Session hijacking
- DoS

**Defenses**
- Intrusion detection
- Cryptography
- Firewalls
- **Secure protocols**
  - PGP
  - SSL
  - IPsec (and VPN)
  - WEP

**Session hijacking**

- Take control of one side of a TCP connection
- Marriage of sniffing and spoofing

**Session hijacking: The details**

- Attacker is on segment where traffic passes from Alice to Bob
  - Attacker sniffs packets
  - Sees TCP packets between Bob and Alice and their sequence numbers

- Attacker jumps in, sending TCP packets to Bob; source IP address = Alice's IP address
  - Bob now obeys commands sent by attacker, thinking they were sent by Alice

- Principal defense: encryption
  - Attacker does not have keys to encrypt and insert meaningful traffic
### Session Hijacking Limitation

**Alice**

1. Weird ACK # for data never sent
2. To resync, Alice sends segment with correct seq #

**Attacker**

1. Bob is getting segments from attacker and Alice. Source IP address same, but seq #s different. Bob likely drops connection.

**Attacker's Solution:**
- Send unsolicited ARP replies to Alice and Bob with non-existent MAC addresses
- Most systems accept replies, overwrite IP-to-MAC ARP tables
- Alice's segments will not reach Bob and vice-versa
- But attacker continues to hear Bob's segments, communicates with Bob

### Session Hijacking Tools:

- **Hunt**
  - Provides ARP poisoning
- **Netcat**
  - General purpose widget
  - Very popular

### Topics Covered in Course

**Attacks and hacker tools**
- Reconnaissance
- Network mapping
- Port scanning
- Sniffing
- IP address spoofing
- Session hijacking

**Defenses**
- Intrusion detection
- Cryptography
- Firewalls

**Secure protocols**
- PGP
- SSL
- IPsec (and VPN)
- WEP

**Before talking about defenses, need to look at network from attacker's perspective**

### Denial-of-Service

- Prevent access by legitimate users or stop critical system processes
  - Remotely stopping/crashing services
  - Remotely exhausting resources
    - Overwhelming connection queue with SYN flood
    - Overwhelming communications link with SYN flood
**Interlude: IP datagram format**

<table>
<thead>
<tr>
<th>Field</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Differentiates between protocols</td>
</tr>
<tr>
<td>total length</td>
<td>Total length of the datagram</td>
</tr>
<tr>
<td>header length</td>
<td>Length of the header portion</td>
</tr>
<tr>
<td>len</td>
<td>Length of the data portion</td>
</tr>
<tr>
<td>id</td>
<td>Identifier for fragmentation</td>
</tr>
<tr>
<td>flags</td>
<td>Flags: fragment offset, more fragments, don't fragment, don't fragment</td>
</tr>
<tr>
<td>options</td>
<td>Options field for additional data</td>
</tr>
<tr>
<td>data</td>
<td>Payload data from the upper layer protocol</td>
</tr>
<tr>
<td>checksum</td>
<td>Checksum for error detection</td>
</tr>
<tr>
<td>ttl</td>
<td>Time to live: number of hops remaining before discarding the datagram</td>
</tr>
<tr>
<td>source ip</td>
<td>Source IP address</td>
</tr>
<tr>
<td>destination ip</td>
<td>Destination IP address</td>
</tr>
</tbody>
</table>

**DoS: crashing services with malformed packets**

- **Land**: Sends spoofed packet with source and dest address/port the same
- **Ping of death**: Sends oversized ping packet
- **Jolt2**: Sends a stream of fragments, none of which have offset of 0. Rebuilding consumes all processor capacity.
- **Teardrop, Newt, Bonk, Syndrop**: Tools send overlapping segments, that is, fragment offsets incorrect.

**DoS: Overwhelming connection queue with SYN flood (1)**

- **Recall**: Client sends SYN packet with initial seq. number when initiating a connection.
- **TCP on server machine**: Allocates memory on its connection queue, to track the status of the new half-open connection.
- **For each half-open connection, server waits for ACK segment, using a timeout that is often > 1 minute**
- **Attack**: Sends many SYN packets, filling connection queue with half-open connections.
- **When connection queue is exhausted, no new connections can be initiated by legit users.**

**Example**

- **4000 byte datagram**
- **MTU = 1500 bytes**

**One large datagram becomes several smaller datagrams**

- Length = 4000, ID = x, frag flag = 0, offset = 0
- Length = 1500, ID = x, frag flag = 1, offset = 0
- Length = 1500, ID = x, frag flag = 1, offset = 1480
- Length = 1040, ID = x, frag flag = 1, offset = 2960

**Patches fix the problem, but malformed packet attacks continue to be discovered.**

**Need to know of open port on victim’s machine: Port scanning.**
**DoS: Overwhelming connection queue with SYN flood (2)**

**Amateur attack:**
- Alice

**Expert attack:**
- Use multiple source IP addresses, each from unresponsive addresses.

**DoS: Overwhelming communications link with SYN flood**

- Fill victim’s access link with SYN packets (inbound) and SYN-ACK packets (outbound).
- Attacker must have more bandwidth than victim.
  - If victim is connected with 1.544 Mbps link, attacker must be able to send 1.544 Mbps of traffic.

**SYN flood defenses**

- Lot’s of bandwidth
- Multiple ISPs for connectivity
- SYN-flood resilient OSES:
  - Large connection queue
  - Lower the timeout value for terminating half-open connections

**SYN flood defense: SYN cookies (1)**

- Defense for overwhelming connect queue
- When SYN segment arrives, host B calculates function (hash) based on:
  - Source and destination IP addresses and port numbers, and a secret number
- Host B uses resulting “cookie” for its initial seq # (ISN) in SYN-ACK
- Host B does not allocate anything to half-open connection:
  - Does not remember A’s ISN
  - Does not remember cookie
**SYN flood defense: SYN cookies (2)**

If SYN is legitimate
- Host A returns ACK
- Host B computes same function, verifies function = ACK # in ACK segment
- Host B creates socket for connection
- Legit connection established without the need for half-open connections

If SYN-flood attack with spoofed IP address
- No ACK comes back to B for connection.
- No problem: B is not waiting for an ACK.

---

**Smurf attacks**

- Idea: Attacker instructs all the machines of a 3rd-party segment to flood a victim.
  - Use 3rd-party’s bandwidth to overwhelm victim’s bandwidth
- Third party segment: “smurf amplifier”

---

**Smurf attack: example**

- Attacker sends ping packet to broadcast address of segment (for 132.65/16 segment, send 132.65.255.255)
- Source IP address of ping packet = victim’s IP address
- Ingress router receives broadcast ping, sends broadcast into segment

---

**Distributed DoS: DDos**

Attacker takes over many machines, called "zombies". Potential zombies are machines with vulnerabilities that can be taken over with buffer-overflow attack.

Zombie processes wait for command from attacker to flood a target.
**Variations on DDoS**

Group of zombies can:
- Stealthy port scan or network mapping
  - Each zombie sends only a few packets
  - Difficult to detect
- Distributed password cracking

**Defenses against DDoS**

- Don’t let your systems become zombies
  - Keep systems patched up
  - Employ egress anti-spoof filtering on external router.
    - Most DDoS attacks use spoofed packets
- Adequate bandwidth, redundant ISPs
- Rapid detection with IDS tools
  - And be able to quickly reach incident response people at your ISP
  - ISP can employ upstream filters
    - Can be tricky since you want to let good guys access your systems

**DNS attacks**

- DNS was designed when security wasn’t an issue.
- What about now?
  - From your home you telnet to utopia.poly.edu, and you get a login prompt. Are you really connected to utopia?
  - Man in the Middle attack is possible by poisoning DNS caches
  - Most ISP’s spam mail blockers use DNS names for blocking mail from rouge sites.
  - Web browser/ISP parental control mechanism work using DNS names
  - Content distribution networks (such as Akamai) works using DNS names

**DNS: redirecting to client**

1. Client sends DNS query to its local DNS server; sniffed by attacker
2. Attacker responds with bogus DNS reply; spoofed IP address
DNS redirection: variation

Same as before, except attacker is sitting near server side.

Poisoning local DNS server (1)

1) Attacker queries local DNS server
2) Local DNS makes iterative queries
3) Attacker waits for some time; sends a bogus reply setting itself as authoritative server for poly.edu.

Poisoning local DNS server (2)

DNS response can provide IP address of malicious server!

DNS Poisoning (3)

- Cache poisoning works because there is no authentication for replies
  - only IP address can be used for authentication.
  - IP addresses can be spoofed
- even easier to do with UDP
  - TCP handshake provides some authentication