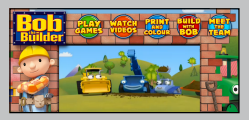
SECURITY (COMP0141): NETWORK SECURITY

UCL

HOW DOES THE INTERNET WORK?

goal: get Alice to that website!



http://me.bob.com/hi.html

Remember back in Week 2 we covered the basics of the internet, which was essential to understanding the role of cryptography in supporting HTTPS

HOW DOES **HTTP** WORK?

goal: get Alice to that website!

`http://me.bob.com/hi.html`

Really though, we didn't learn how the internet worked at all, but rather mostly just HTTP (and then HTTPS)

INTERNET PROTOCOL SUITE

Application layer: SMTP, FTP, SSH, HTTP, etc.

Transport layer: host-to-host communications (UDP, TCP, etc.)

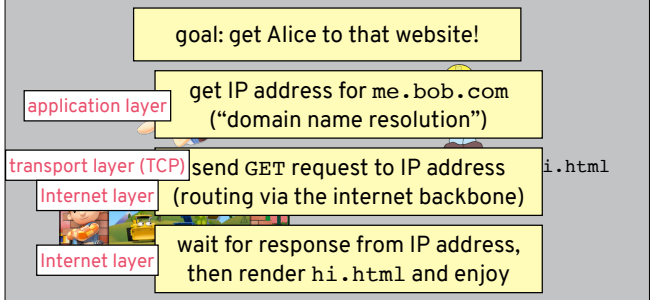
Internet layer (IP): End-to-end routing of data packets

Link layer: Transmission of data within a local network (Ethernet)

Physical layer: Transmission of raw bits over a physical link

HTTP lives at the highest layer of abstraction in terms of Internet protocols

HOW DOES THE INTERNET WORK?



Some of the other things we described, like routing, live at different layers of abstraction

---

HOW DOES **TCP/IP** WORK?

Application layer: SMTP, FTP, SSH, HTTP, etc.

Transport layer: host-to-host communications (UDP, TCP, etc.)

Internet layer (IP): End-to-end routing of data packets

Link layer: Transmission of data within a local network (Ethernet)

Physical layer: Transmission of raw bits over a physical link

We're going to go back and see more detail on these lower levels of abstraction, looking in particular at TCP/IP. If you want to learn about the stuff at even lower levels then you should take Networked Systems next year

STEP 1: FIND CONTENT HOST

Remember that the first step was performing something called domain name resolution using DNS

DOMAIN NAME SYSTEM

FAQs

q: do we really do this every time we go to a website?

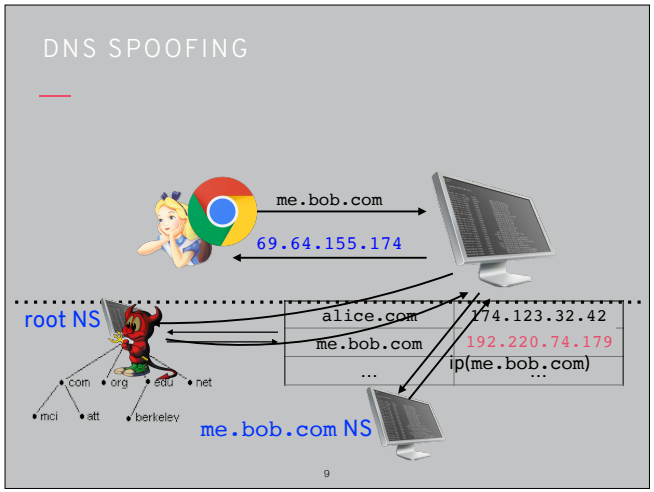
a: no! DNS results are cached by your browser.

DNS responses and negative queries are cached, and cached data expires according to TTL (time to live) provided by owner of data

These results are also cached by the resolver itself

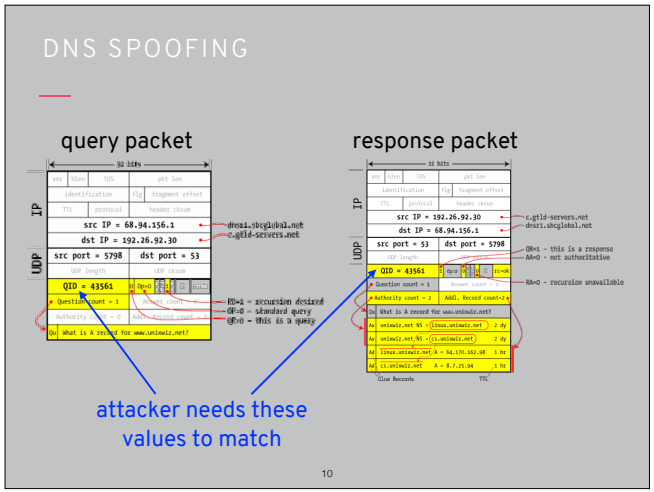
This opens up the potential for DNS cache poisoning because the NS does not validate that DNS entries are from authoritative sources

Remember also that DNS results get cached, and in fact name servers also cache their results. This creates the potential for the first attack we'll look at, which is DNS cache poisoning



The attacker doesn't have to compromise the name server, just send them IP address information for a domain they don't control, which works because no one is checking the source of this information

---

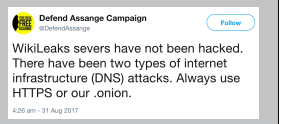


In a little more detail, DNS queries have QID that ties response back to query, means attacker may have to guess this value

DNS SPOOFING

Many examples of this being done in practice:

- 2000: `hilary2000.org` sent to `hilaryno.com`
- 2004: Google and Amazon sent to an online pharmacy
- 2016: all 36 of a Brazilian bank's domains sent to phishing sites (pharming)
- 2017: Wikileaks visitors sent to attacker-controlled page

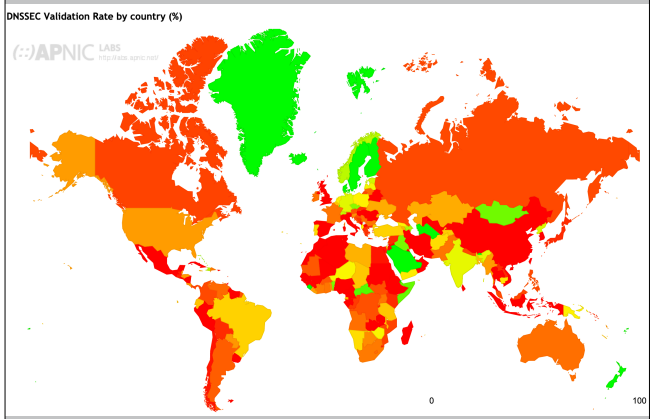


Solutions include:

- Randomising source port (more randomness for QID)
- Ignoring unnecessary responses
- DNSSEC: authenticate responses with digital signature

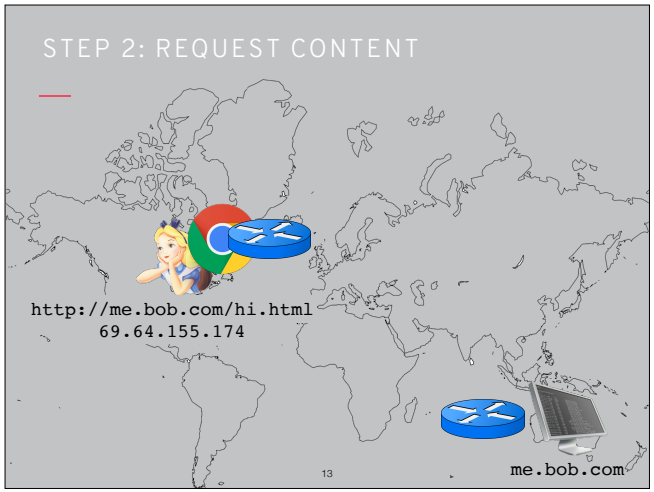
11

Remember we saw pharming in Week 7

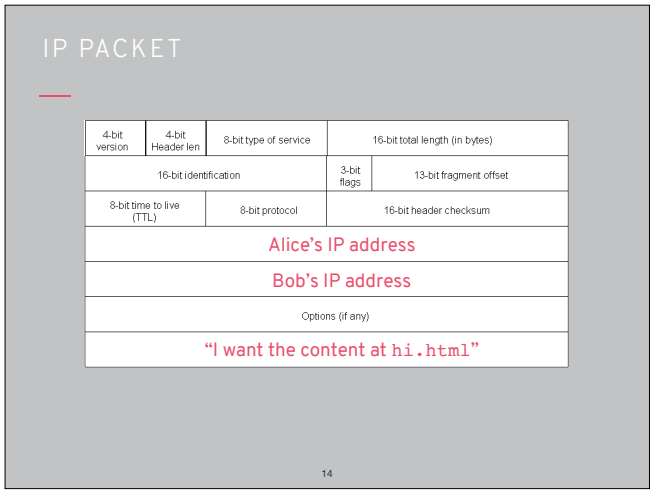


https://stats.labs.apnic.net/dnssec

Some countries are doing very well with adoption of DNSSEC but most are not

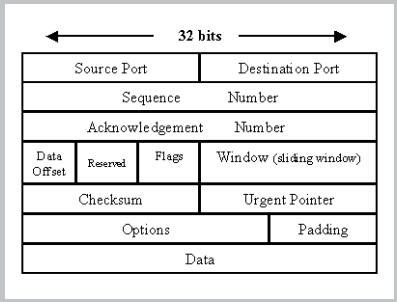


So after we get an IP address, remember that the next step is routing, which means getting more into TCP/IP



Remember that packets are what get sent around the network, in fact these are IP packets that are structured according to the Internet Protocol

TCP PACKET



15

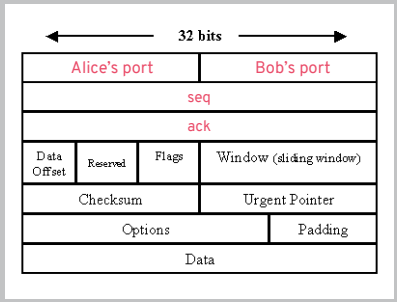
Often what is contained within an IP packet is a TCP packet, this nesting technique is often called encapsulation (https://en.wikipedia.org/wiki/Encapsulation_(networking)), we also saw it with the packets for DNS

TCP

Used to establish a bi-directional **stateful** session between two endpoints identified by their IP address and **port**

16

TCP PACKET



TCP

Used to establish a **stateful** bi-directional session between two endpoints identified by their IP address and **port**

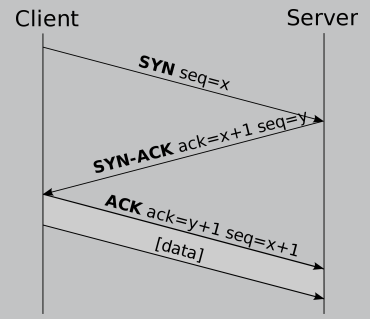
- 22: SSH (remote access)
- 53: DNS
- 80: HTTP
- 443: HTTPS

Packets can contain special **flags**:

- SYN: I want to start a connection
- FIN: I want to close a connection
- ACK: I got your last packet

Different ports are used for different services, here are some examples of very common ones. There are also common flags that can be used to indicate certain things

TCP HANDSHAKE



19

All TCP connections start with what is called the TCP handshake, which consists of three messages. The state contains the sequence number and the ack number, and these act as a weak form of authentication

TCP

Used to establish a **stateful** bi-directional session between two endpoints identified by their IP address and **port**

- 22: SSH (remote access)
- 53: DNS
- 80: HTTP
- 443: HTTPS

Packets can contain special **flags**:

- SYN: I want to start a connection
- FIN: I want to close a connection
- ACK: I got your last packet

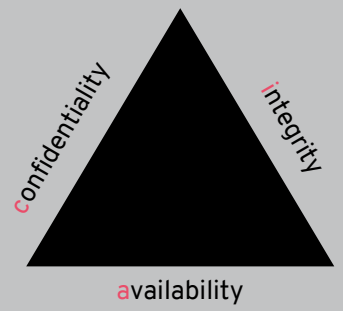
TCP/IP trust model has evolved:

- 1970s: trusted network and trusted hosts
- 1980s: hosts may be compromised
- today: network may be compromised too

20

Trust assumptions have changed over the decades, started out very strong and are now as minimal as possible (remember that this is a good thing from the standpoint of trust models, want weakest possible assumptions)

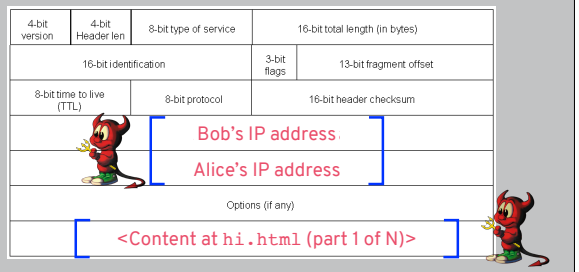
CIA TRIANGLE



21

Let's revisit the notions of CIA in the context of network security

CONFIDENTIALITY



anyone can read your web traffic

anyone can see who you're talking to

22

Although we saw how HTTPS can be used to hide the content of IP packets, the default is that everything is in the clear, so there is no confidentiality

IP SPOOFING

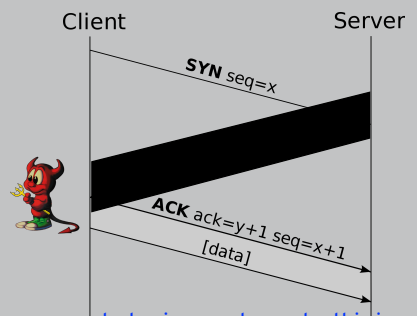
4-bit version	4-bit Header len	8-bit type of service	16-bit total length (in bytes)	
16-bit identification			3-bit flags	13-bit fragment offset
8-bit time to live (TTL)		8-bit protocol	16-bit header checksum	
Professor Evil's ~~Alice's~~ IP address				
Bob's IP address				
Options (if any)				
"I want the content at `hi.html`"				

anyone can impersonate you (or anyone else)

23

Remember, there is no authentication on the Internet so it is easy to spoof IP addresses (i.e., pretend something came from a place even if it didn't)

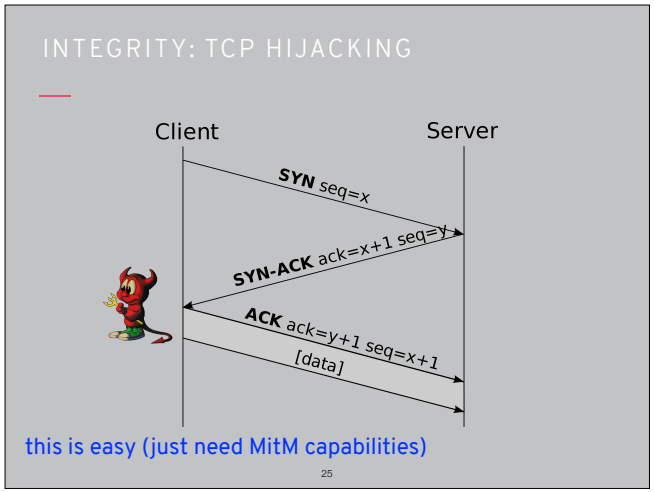
INTEGRITY: TCP SPOOFING



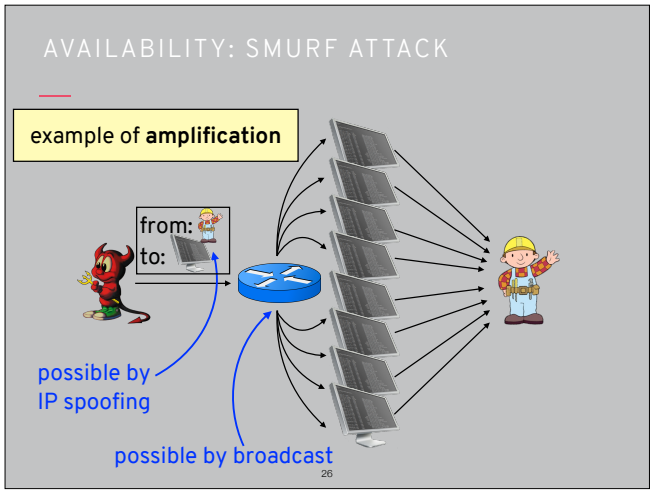
if ack,seq generated using weak crypto, this is possible
(and both are only 4 bytes so can be brute-forced)

24

TCP spoofing is more complicated because the two endpoints maintain shared state, so attacker would need to guess sequence number without seeing SYN-ACK response

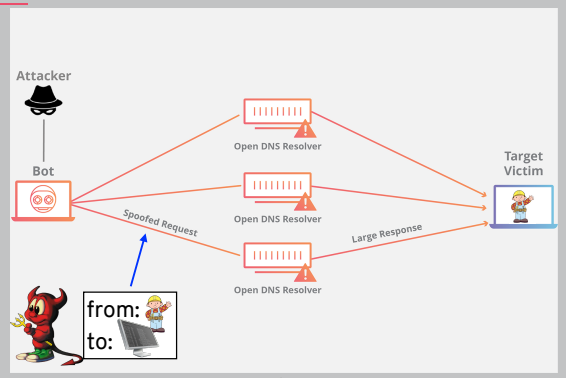


On the other hand if the attacker can observe every message then they can learn both ack and seq and perform what's known as session hijacking



We've already seen one example of an attack on availability, in the form of the Smurf attack, which is an example of a more general technique called reflection. This uses a special type of packet though (an ICMP echo request), so let's see another example

AVAILABILITY: DNS AMPLIFICATION



27

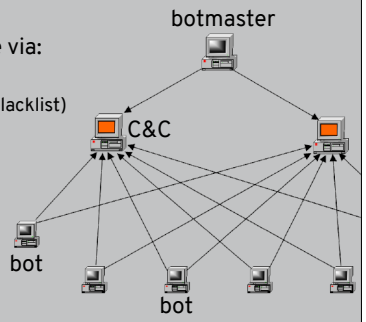
DNS amplification is similar, but attacker pretends to be Bob asking for a lot of information, like in a memcaching attack. The request is small but the response is very large so again this is an example of amplification/reflection

---

AVAILABILITY: BOTNETS

communication takes place via:
- IRC (easy to infiltrate)
- proprietary channels (easy to blacklist)

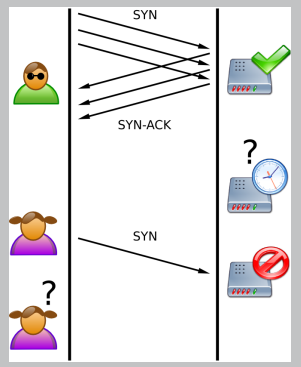
structure uses:
- multiple tiers (expensive)
- p2p (easy to infiltrate)
- fast flux/domain flux (hard!)



28

Remember when we talked about availability, we also saw how botnets use fast flux and domain flux

AVAILABILITY: SYN FLOOD



A TCP SYN flood creates many half-open sessions by sending the initial SYN packet but not the later ACK one. This overwhelms the number of TCP sessions a server can support and prevent legitimate users from connecting

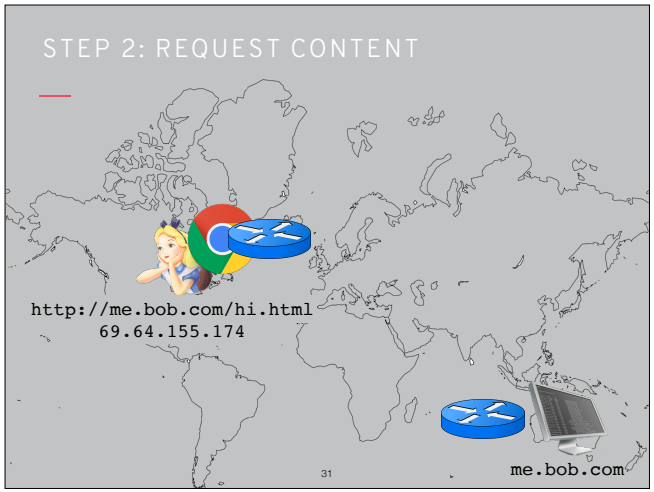
AVAILABILITY: SYN FLOOD

Several proposed countermeasures:

- Filtering (don't allow so many connections from same place)
- Reducing SYN-RECEIVED timer (don't wait so long to close connection)
- SYN cookies
- Firewalls and proxies

SYN cookies:

- Let t be slow timestamp (e.g., changes every minute)
- Let m be maximum segment size (MSS)
- Let s = H(IP addresses, ports, t)
- Initial seq ("SYN cookie") = 5 bits t + 3 bits m + 24 bits s
- This seq+1 is sent as ack in TCP ACK
- Server can check t within range, compute s and check equal



So what started this whole investigation of TCP was the idea of routing, but so far we have said very little about it – this is really the IP part of things. That's what we'll see next time!