COMP3310/6331 – #21

Security

Dr Markus Buchhorn: *markus.buchhorn@anu.edu.au*

Security at what layer?

- Spans all layers
- Every layer provides <u>opportunities</u> and <u>risks</u>
- From the application down to the physical
- Various security designs, for a variety of threats
  - It's more than just cryptography

Security?

- In the eye of the beholder
- Need to understand a range of properties
- Threat model:
  - What are the dangers?
  - What are the capabilities of the ‘attacker’?
  - What are the probabilities, impacts and costs?
- All help to <u>assess</u> the risk – and the <u>effort</u> to mitigate it
  - Remove a risk, or deal with the consequences?

Example threats and levels

- Note: focussing on network security, rather than application security
  - But each is a vector into the other!
  - The edge is not very clear
  - And scale can be tiny/targeted to huge/widespread

- Eavesdropping: *Intercept messages, gain content (passive)*
- Intrusion: *Compromise device, modify messages (active)*
- Impersonation: *Identity fraud, gain content, modify messages (active)*
- Extortion: *Disrupt services (active)*

Vulnerabilities

- Attack surfaces
  - How many places are you vulnerable?
  - How do you know??
  - E.g. Use of cloud services for smart devices

https://mjg59.dreamwidth.org/40397.html
"I bought some awful light bulbs so you don't have to"

- Single points of failure
  - Can I knock you out at a single device?
  - Routers, servers, directories, databases, file, …

Security = risk management

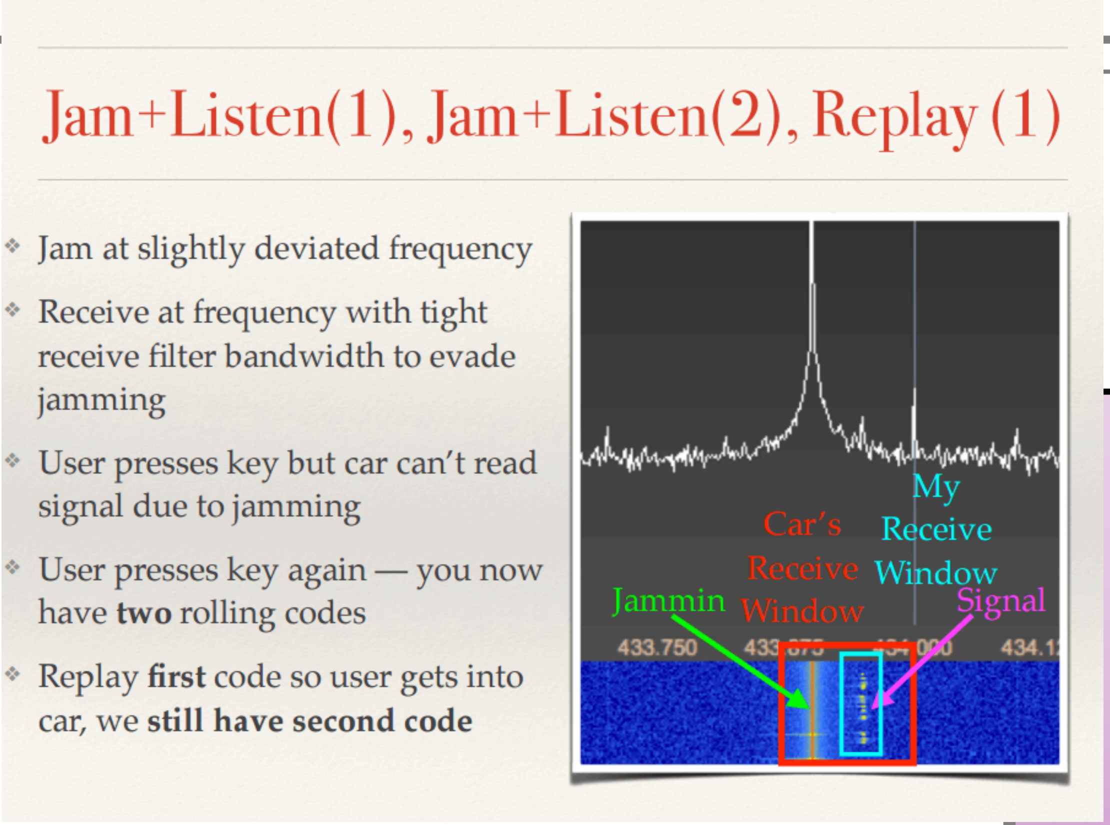
- Can never be perfect – cannot prove an absence
- Ensure security model to minimise probabilities
- Only as secure as the weakest link:
  - <u>Design flaws</u>
    - Poorly thought through
    - Law of unintended consequences (multi-component systems)
      - The Internet is the world's largest multi-component system…
  - <u>Code flaws</u>
    - Always one more bug
  - <u>Somebody else's flaws…</u>
    - Human behaviours
      - Accidentally, deliberately

e.g. Replay attacks

- Samy Kamkar – Hackaday Supercon
- Jam and Listen
  - Steals codes without alerting car/user
- Pick up short-range car-signal for nearby keyfob
  - retransmit to friend near car-owner, return response, car unlocks!
- Find owner: Send hunt signal into room and listen for responses!

https://www.youtube.com/watch?v=RpD-yMcg4P4

https://www.youtube.com/watch?v=1RipwqJG50c



e.g. Wifi is ‘more’ secure now…?

- Old WEP Cryptography – really weak – easy to snoop and decrypt
- 802.11i
  - Nearly impossible to brute-force decrypt on today’s computers
- Removed one threat, but e.g
  - Could have configured a guessable SSID/key – gets attacker on the WLAN
  - Could have wired LAN access to devices on WLAN – gets attacker onto LAN/WLAN
  - Physical access to network devices (access points)
    - People have tapped the chips on AP boards
- All lead to someone accessing the network, and listening to (some) traffic
  - And sending, possibly

Naïve Internet designs

- Security has been added on over the years
  - Many protocols came from a smaller, friendlier network community
    - DNS, DHCP, HTTP, …
    - Clients/servers had prior relationships
    - No concept of identity in most protocols
      - And which identity?
- Earlier designs never considered the value attached to the network
    - Banks, businesses, factories, power stations, government agencies, control systems, …
- Significant effort to retrofit security
  - Or completely redesign with security in mind
  - Or use other mechanisms to provide security

Basic assumptions

- There will always be villains, in various forms and paygrades
  - And they know nothing initially – so probe. All the time.

- All physical links can be interfered with
  - snooped, misdirected, cut, added, …

- <u>You can’t trust packets; headers nor payloads!</u>
- Protocol designs are public and have many holes
- Security ‘on the wire’ can be undermined by security ‘on the host’
- Technology is easy to hack, and people are even easier

Crypto-graphy…(“Hidden writing”)

- A common first point for security
  - For data ‘in motion’ (travelling over networks)
  - For data ‘at rest’ (in applications)

- Cryptography
  - Encrypt information
  - Make it computationally infeasible (too much time) to decrypt
  - But it has an ‘edge’, where it stops

- Cryptanalysis
  - Try to decrypt information

Encryption

- <u>Not just</u> for preventing eavesdropping (confidentiality)
- Can also
  - Confirm messages came from device you expect
  - Confirm remote party is who they say they are
  - Validate message has not been tampered with
- Remarkably easy to develop poor encryption
  - Many half-baked attempts over many years
  - Stick with well-known platforms/systems
    - And try to use them well!
    - And still be wary

Confidentiality

- Encryption to ensure messages can’t be read while travelling
  - Goal: send a private message from A to B
  - Threat: (Passive) villain-in-the-middle reads message along the path
  - “Application” end-points en-/decrypt messages
- Two common approaches:
  - Symmetric (shared key)
  - Asymmetric (public/private key)

Symmetric vs Asymmetric

- Shared secret crypto
  - Both parties have the **same key**, use it to encrypt/decrypt messages
    - Same algorithm used at both ends (e.g. AES)
    - Assume attacker knows the algorithm
    - Sharing the key is a weakness
- Public Key crypto
  - **Key pair** (public/private);
  - Owner (only) holds private key, anyone can/should have public key
  - Encryption through expensive mathematics (e.g. RSA)
    - “Only” private key can decrypt public-key-encryption, and v.v.
  - *Public Key Infrastructure (PKI)*
- Rule: Want network to carry ciphertext (encrypted messages) only.

Key distribution and performance

- Trade-off: which approach?
- **Symmetric**
  - Encryption is lightweight/fast – great for high data rates
  - Key sharing is hard:
    - Need to get (different) key to everyone who needs/deserves it
    - For every new conversation
- **Asymmetric**
  - Encryption is heavy/slow – not for general use
  - Key sharing is easy

  - But also need to know you can trust the public key
    - Bind an identity to the key
    - Needs a directory service (see *certificates*)

Best of both?

- Use <u>public key</u> encryption to send a <u>shared</u> key

- Use <u>shared key</u> for encrypting further communication
  - Ensure confidentiality

- This shared key is a <u>**session-key**</u>
  - Short-term use, encrypt each <u>packet</u>
  - As big as you need it to be
  - Can generate a new one each time

Authentication and Integrity

- Confidentiality is great against passive villains
  - They can't read the packets, and you can authenticate source

- Active villains (intruder) may still tinker with the **message** en-route
  - Incorrect, misleading, broken message gets through
  - Corrupt, replay, reorder packets
    - *WAIT DO NOT STOP – STOP DO NOT WAIT*

- Need **message integrity**
  - Use session-key (established through PKI)
  - Calculate a summary of the **message** (signature, message digest, hash)
    - And encrypt that with session key or private key

Freshness

- Villain can store (encrypted) messages,
- and send them again and again - later
  - E.g. ‘transfer $1000 to X’
  - E.g. ‘set password = ABCDEFG’
  - No matter how well encrypted, as long as <u>within timeframe of session key</u>
- Replay attack
- Easy fix: include timestamp (or other ‘nonce’) in the message/signature
  - Before encryption
  - (Application requirement, not part of crypto)

Applying cryptography

- Each application can now build their own
  1. Confidentiality
  2. Authentication
  3. Integrity
  4. *Freshness*

  into their protocols
- We trust every app developer, every language, every OS to get it right?
- Why not add it to the network?
  - Which layer - Link, Network, Transport?
    - Options for each of them…

Establish a Secure Socket Layer

- SSL (1995/96 SSLv3) – Netscape browser
- Triggered by HTTP ➔ HTTP/S (HTTP over SSL) = **https://**
- Led to generalised *Transport Layer Security* (TLS)
  – V1.0 1999, V1.1 2006, V1.2 2008, V1.3 2018
- No longer specific to HTTP
- Sits "between" Transport and Application – encrypts tcp payloads

DTLS = TLS/UDP

HTTP, …	Application
SSL/TLS	
TCP	Transport
IP	Network

https://www.howsmyssl.com/s/about.html

What do we get?

- SSL/TLS provides
  - Verification of server by client (padlock icon in www)
  - Message exchange,
    - with confidentiality, integrity, authentication and freshness

- Starts with authentication phase
  - To establish encrypted channel and session key(s)
  - Before any single application message (HTTP) is exchanged

- Client needs to authenticate some random new server
  - Network traffic can be spoofed and misdirected
  - Needs server public key, **for sure**… provided by a <u>certificate</u>

Certificates

- **Bind an identity to a public key**
  - Server/service, or person
- Requires somebody authoritative to say so
  - Certificate Authorities (CA)
  - X.509 standard
    - Metadata about identity, plus public key, encrypted with CA private key


I hereby certify that the public key
19836A8B03030CF83737E3837837FC3s87092827262643FFA82710382828282A
belongs to
Robert John Smith
12345 University Avenue
Berkeley, CA 94702
Birthday: July 4, 1958
Email: bob@superdupernet.com

Signed by CA

Public Key infrastructure

- Can't have just **one** Certificate Authority
  - Too busy to deal with the whole Internet
  - Too big to fail
  - Too obvious a target
  - Too easily untrustworthy and a monopoly
- Build a hierarchy
  - One or more competing 'root' CA's
  - That validate/sign delegate CA's
    - That …
      - That …
        - That sign your/your webserver's certificate

Anchoring trust

- Browsers hold certificates for root CAs
  - Around 70 **root** certs in Chrome today – and cache many more
  - That's a lot of trust…

- On SSL/TLS initiation, request server certificate
  - And check its CA signature
    - And check its CA signature
      - And check its CA signature
        - … up to the root signature

- Nice, till somebody gets hacked
  - Private key is exposed
  - Certificate must be revoked
  - PKI has a Certificate Revocation List (CRL) – this does not scale well!

That solves everything?

- Not even close

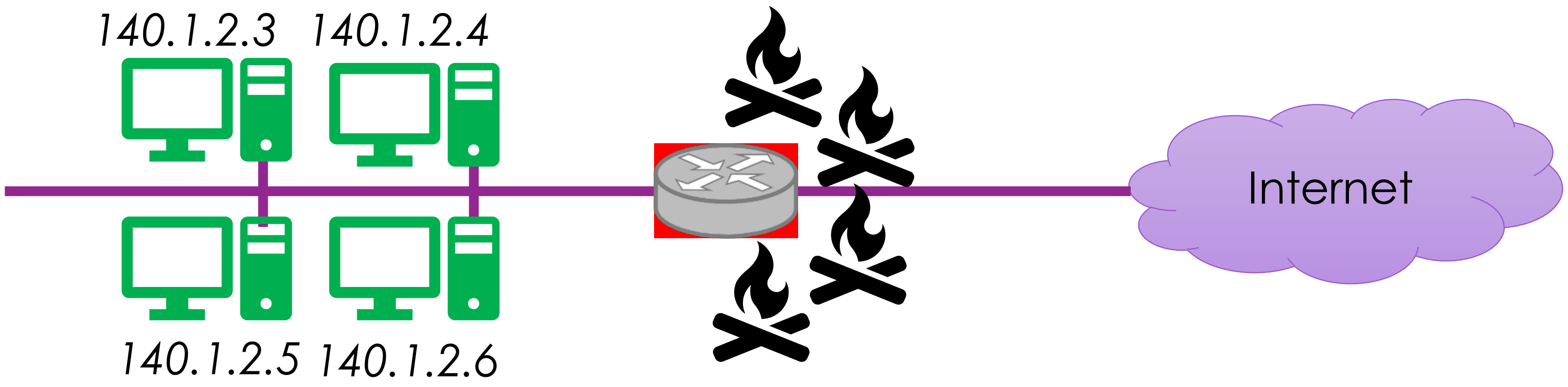
- SSL/TLS: a common security layer between applications and transport
  - And has itself been broken a few (12+) times
  - Code flaws, crypto flaws, interaction flaws with other protocols, spooks, …
  - Not used by all applications, or other layer protocols (DNS, DHCP, BGP, …!)
    - Recall DNSSec discussion earlier

- If you need it, use it – or get a PhD in crypto (and then use it).

- What about other layers?

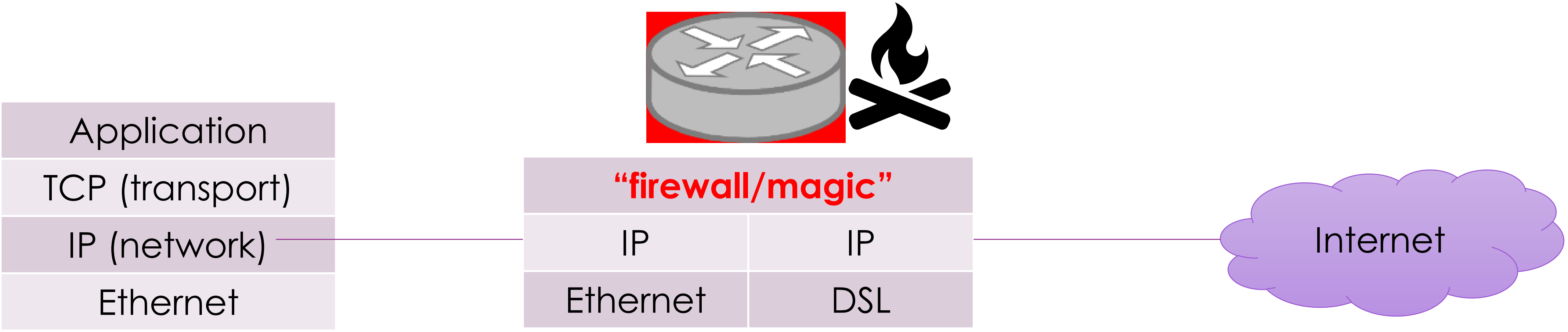
Firewalls

- Edge routers/gateways/processes that (can) explicitly block packets
- Internet:
  - You can send to any host.
  - Any host can send to you!
- Only want to let the nice packets in (and out)



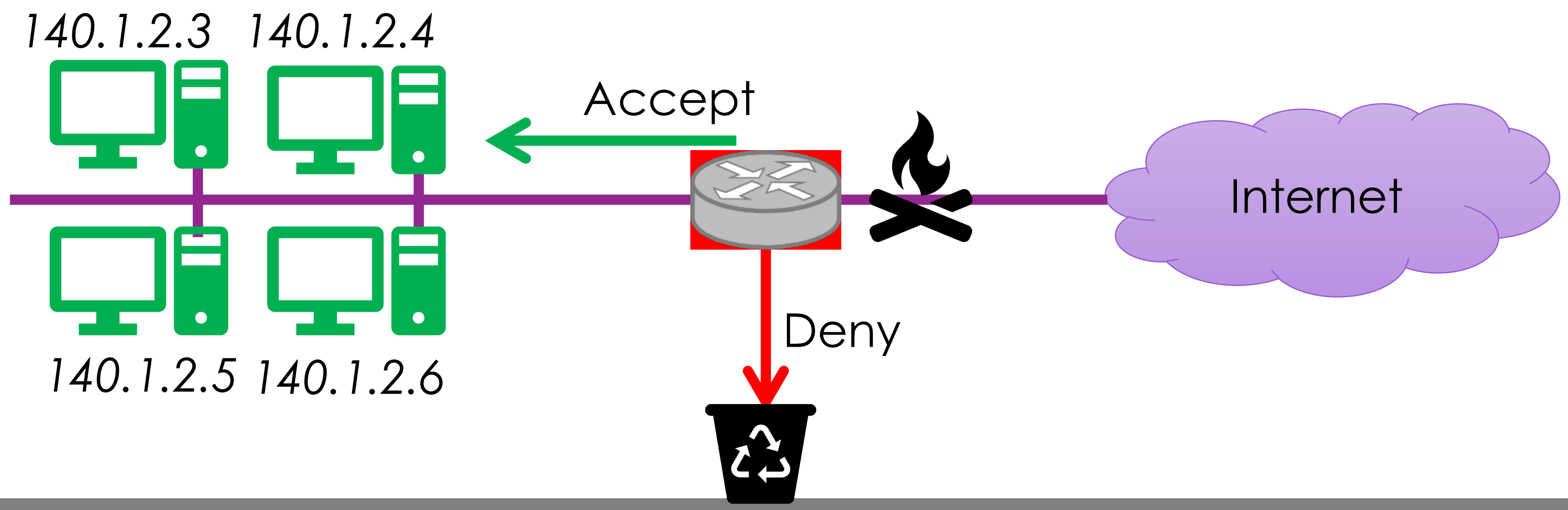
The “middlebox”

Routers (normally) just look at IP – but…



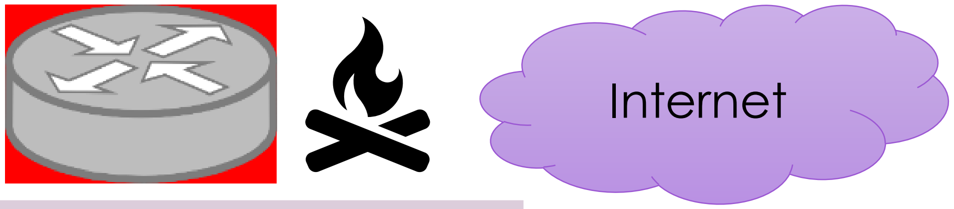
Policy time

- Establish Accept/Deny rules
  - Break applications we don't like
  - Very high level
- Packet filtering is low-level
  - Doesn't look at protocol messages, encrypted traffic, …



Firewalls at different layers

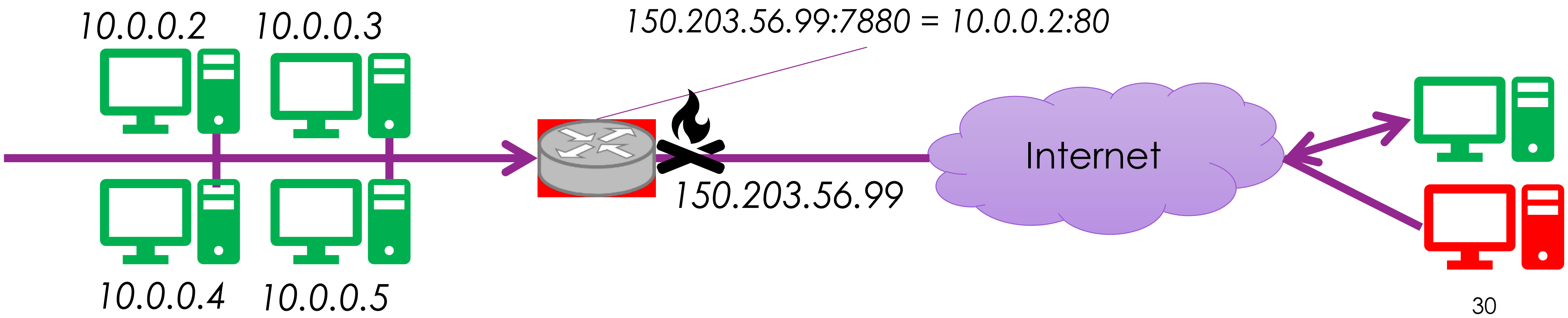
- Basic block – "stateless"
  - No state from one packet to the next
  - Allow/Deny based on IP addresses
  - Allow/Deny based on TCP vs UDP
  - Allow/Deny based on Port numbers
  - E.g. deny port 25 tcp (email)
  - E.g. deny all, allow port 80 tcp (http)
    - Because… people.



"firewall/magic"	
IP	IP
Ethernet	DSL

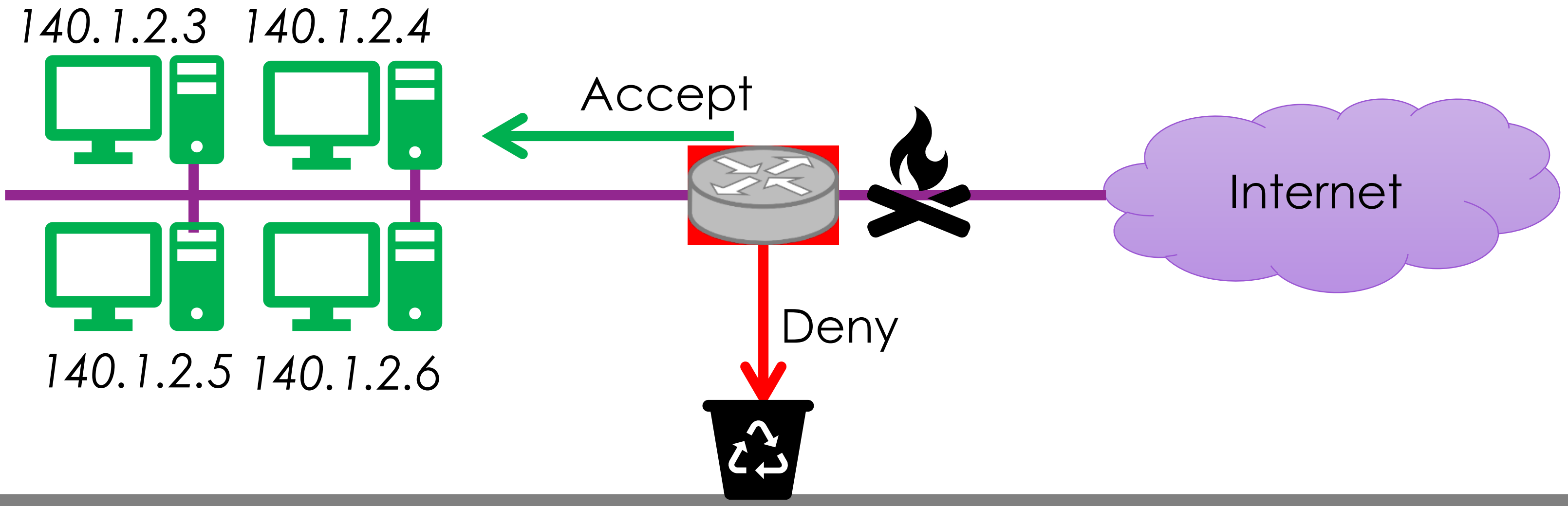
Stateful Firewalls

- Change the rules based on other triggers
  - Track packet <u>flows</u> between internal and external hosts
  - E.g. NAT: Allow inbound TCP from any outside X to our inside Y <u>that initiated a connection to X</u>
    - But timeout after idle…



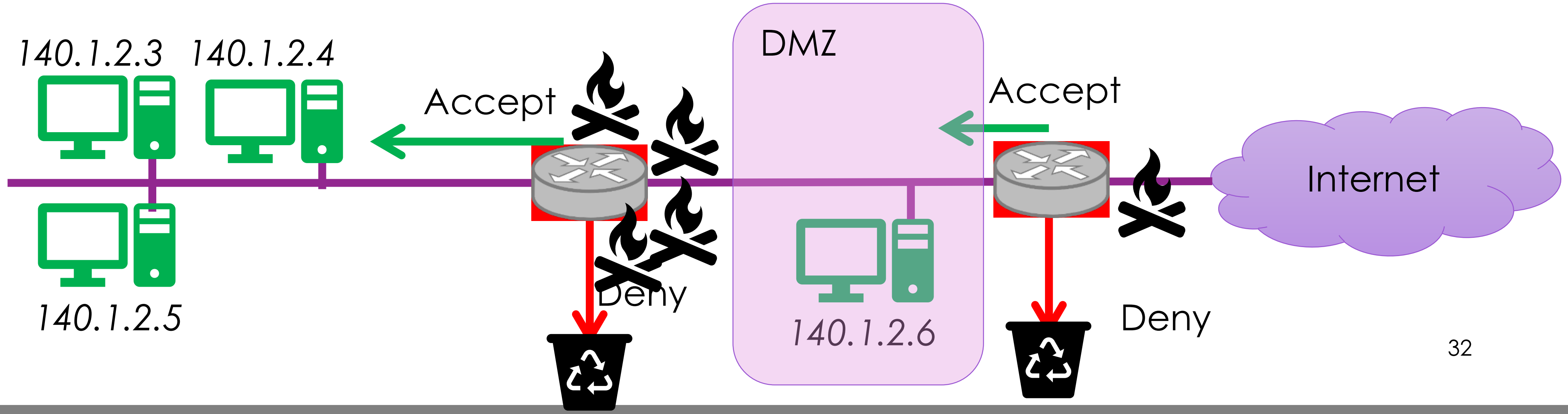
Application firewalls

- **Deep Packet Inspection**
- Understands application protocols
  - Will reassemble messages from packets
  - Understands content
    - E.g. viruses in emails/web pages
- And performance suffers



Firewall deployment

- Sometimes need to protect some devices more than others
  - Internal finance system versus your company web-server
- Two stage firewall: create a ‘demilitarised zone’ (DMZ)



Firewall implementation

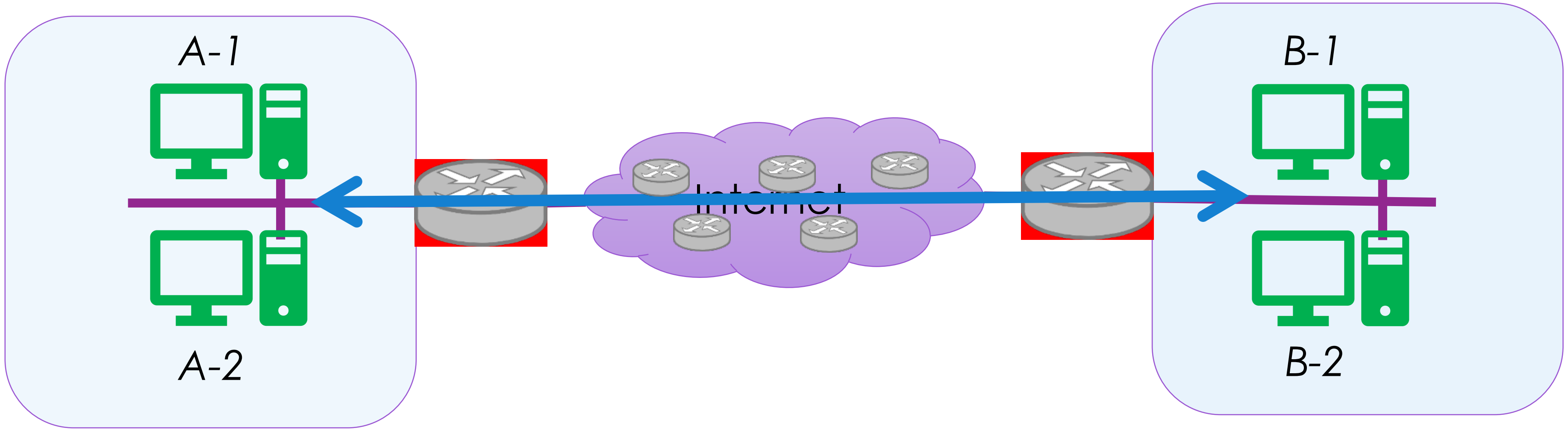
- Dedicated devices
  - Especially Application/DPI Firewalls
- Routers/Modems
- Wireless Access Points
- On hosts, in the Operating System (e.g. Linux IPTABLES)

- Tradeoffs
  - Multiple firewalls = more security **AND** more places to break/slow things
    - More places to maintain and coordinate
    - Protect hosts from each other

Firewalls = Islands of trust

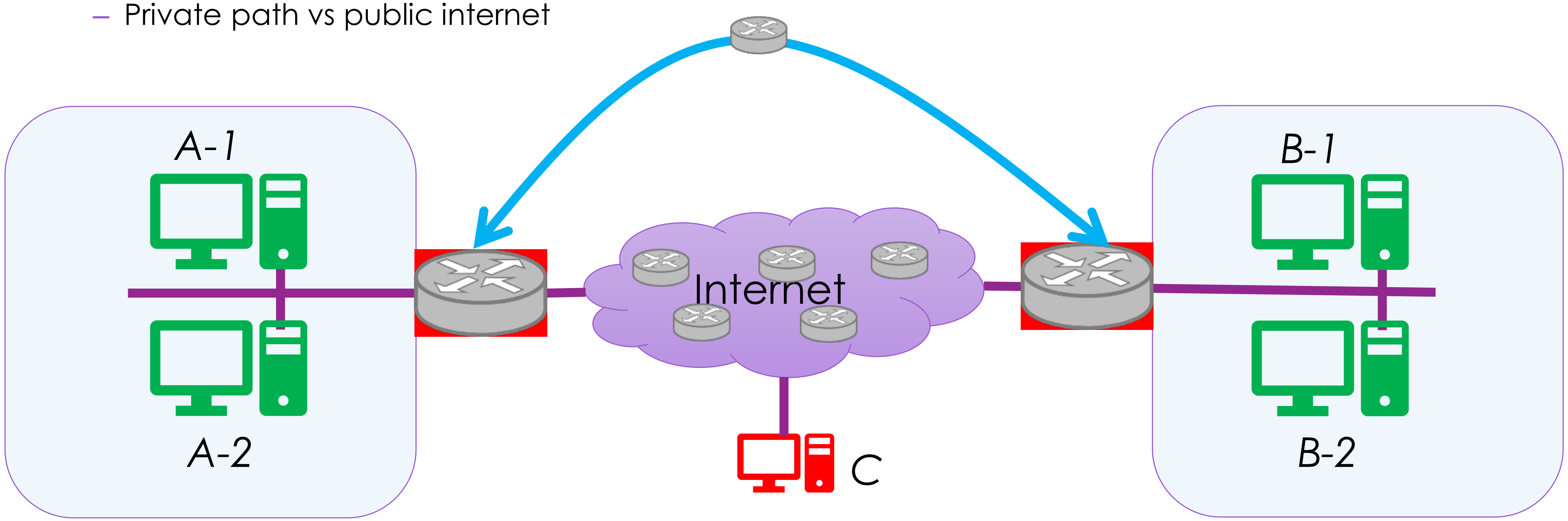
- *SSL doesn't hide everything*
  - Intermediate routers can see the traffic
    - Leak information about activities
  - What about *End-to-End* confidentiality/etc?
    - Host to host
    - subnet to subnet

HTTP, …
SSL/TLS
TCP
IP



Islands of trust

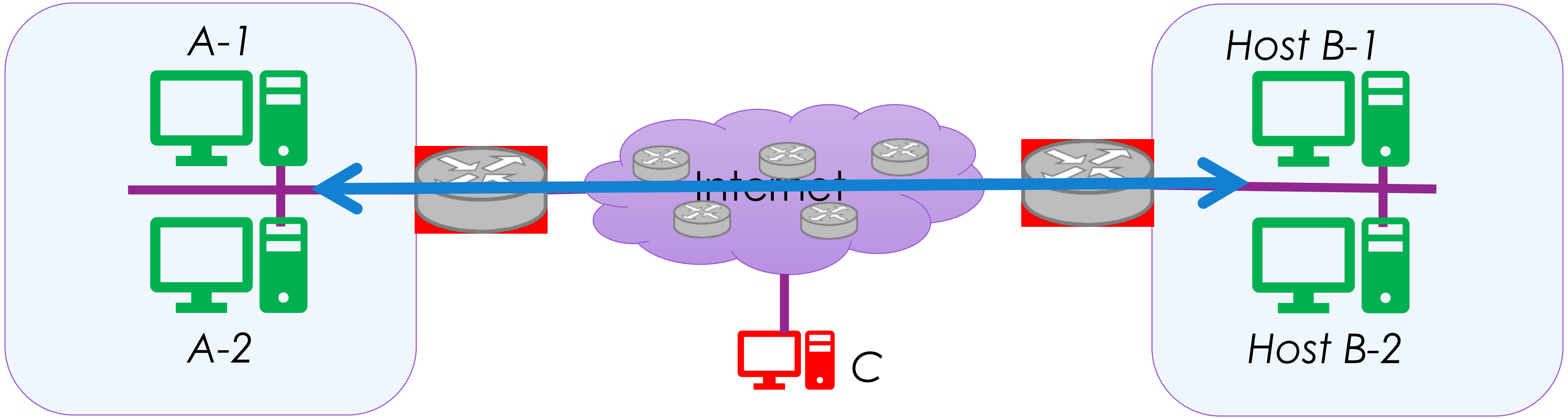
- **Leased lines**
  - Who needs the Internet?
    - Effective – sort of (smaller attack population)
  - But
    - Doesn't scale to buy physical links ($$)
    - Need to manage routing ($$)
      - Private path vs public internet



Islands of trust

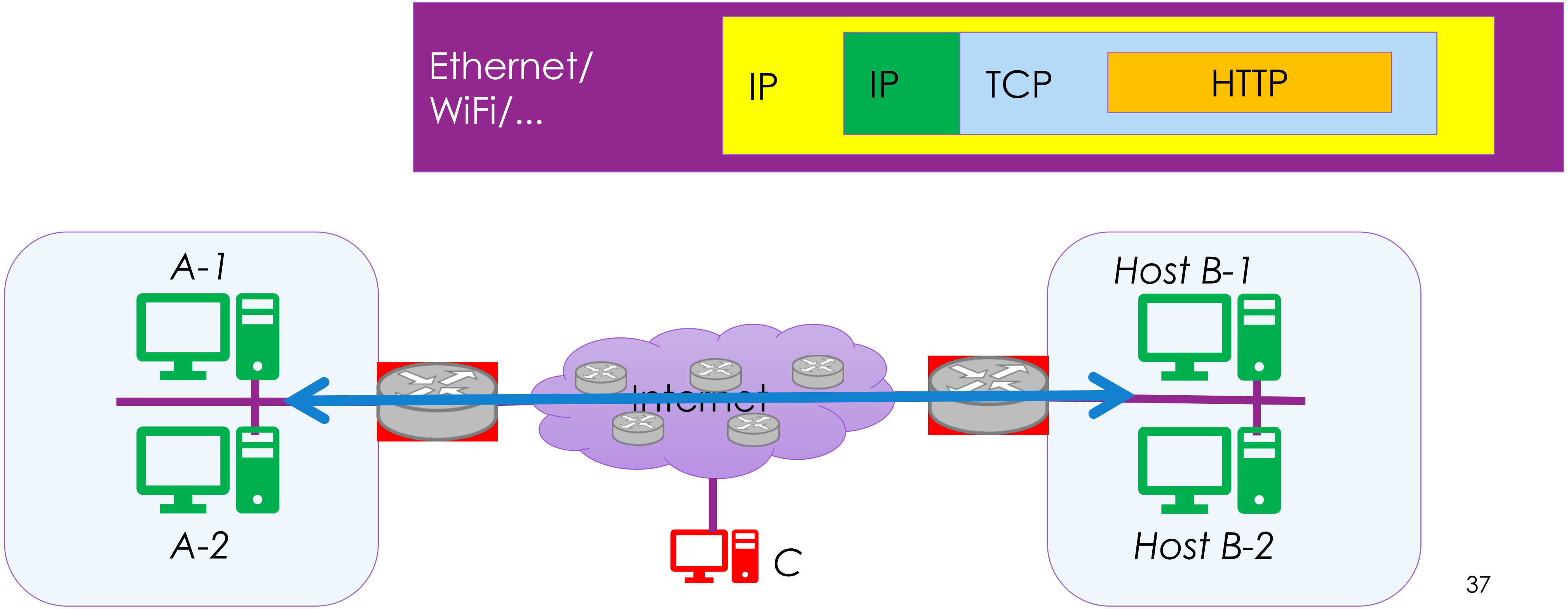
- Can we use the Internet?
  - Need security at the IP layer
- Make a “virtual” leased line
  - <u>*Virtual Private Network (VPN)*</u>

HTTP, …	HTTP, …
SSL/TLS	SSL/TLS
TCP	TCP
IP	Secure IP



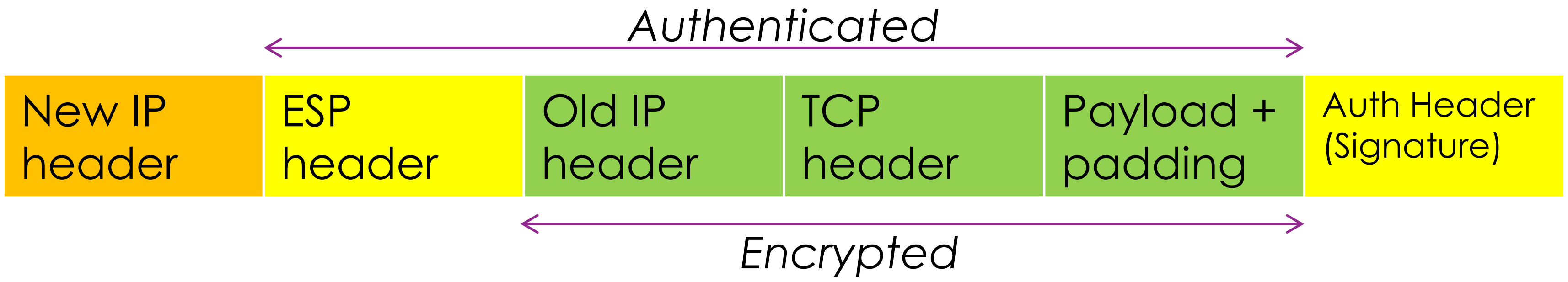
Islands of trust - VPNs

- Tunnel (=encapsulate) IP packets across the Internet (IP in IP)



IP tunnelling security

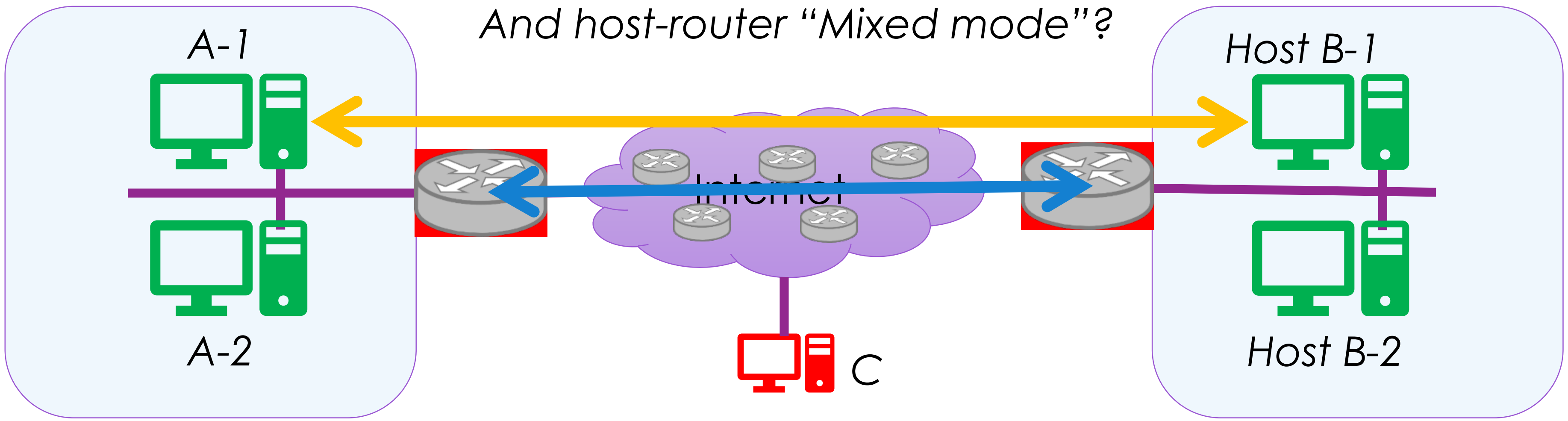
- IP in IP (encapsulation) has **no** protection
  - They're still ordinary packets, just an extra header
  - No *confidentiality*, *authentication*, or *integrity*
- Use **IP Security (IPsec)** to establish secure VPN connections
  - Cryptography at the network layer
  - Keys are exchanged between **endpoints** (can be hosts and/or routers)
  - Packets are encapsulated and encrypted



(ESP = Encapsulating Security Payload)

VPN endpoints

- **Tunnel mode**: router to router (i.e. with forwarding)
  - Connects whole subnets transparently – make them look as one
  - Can be NAT-friendly
- **Transport mode**: host to host (i.e. no forwarding)
  - Different format, only encrypt IP payloads, NAT-challenged.



Aluminium-foil hats vs black hats

- Good encryption is bad national security?
- NSA and others accused of
  - *Modifying Operating System code*
  - *Modifying VPN code*
  - *Modifying encryption algorithms*
  - *Precomputing encryption/hash keys*
  - *Targeting device firmware, motherboards*
  - …
  - It's in their interest!
- Longstanding effort to make IP secure.
  - Performance and deployment issues
  - IPv6 tried to make IPSec compulsory
    - But runs too slow on small (IoT) devices

Address transparency

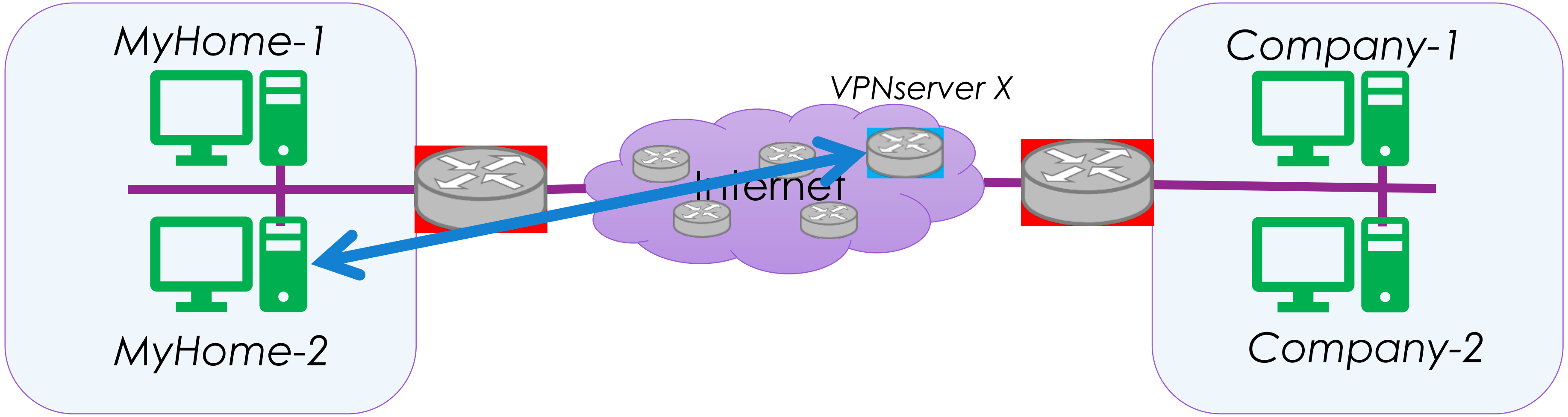
- New packet gets <u>IP address of tunnel endpoints</u>
  - Effectively a new layer, IPsec over IP
  - Can’t identify (original) source/destination
    - Until you come out of the tunnel
  - Good thing? Bad thing?
    - Break firewalls (both for simple IP and packet inspection)
    - Undermine optimal routing
      - Sort of…

HTTP, …
SSL/TLS
TCP
IPsec
IP

New IP header	ESP header	Old IP header	TCP header	Payload + padding	Auth Header (Signature)

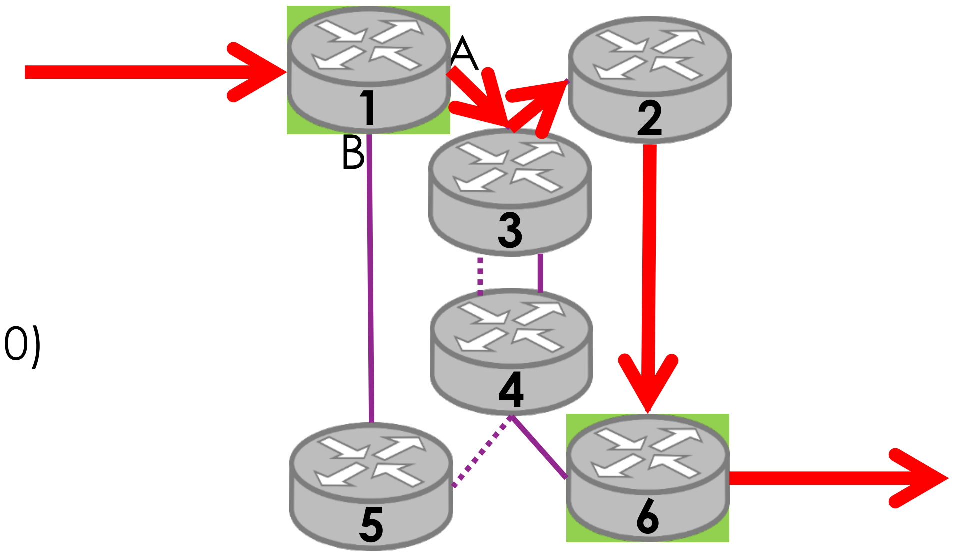
Address opaqueness

- A VPN <u>endpoint</u> is not necessarily the <u>destination</u>
  - Just where the packets 'emerge' from the tunnel
  - 'source' address is the <u>tunnel endpoint</u>
    - Responses need to go back there
  - IP addresses are allocated to geographical regions
    - This has some benefits…



Looks a bit like a circuit?

- Well, yes.
- But only the endpoints are tied down
  - Internet routing handles everything in between
    - Dynamically, politically, …
    - Deals with failover, multi-path, …
  - Packets are labelled with IP addr.
  - Fails when either endpoint dies
- Compare with MPLS
  - Multi-protocol Label Switching (back in T10)
  - Pre-establish the entire path
  - Packets are labelled with a token
  - Fails when any path-element dies



What about physical security?

- If villain has access to the actual links…

- Network Devices
- Copper
- Fibre
- Wireless

Device security

- Switches/routers have physical and virtual interfaces
- Remote access
  - SNMP agents (http admin)
  - Operating systems (telnet/ssh)
  - Port monitors/mirrors
    - Reflects traffic to another port
    - Can't tell from outside
- Physical access
  - Interface rearrangement (denial) or attacks
  - Cable cutting/interference
  - Chip-pin-level snooping

Copper security

- Easy to tap
- Hard to detect
- Actively cut cable
  - Denial of service or Impersonate Device
- Splice cable
  - Eavesdrop what is on the wire
  - Impersonate Device
  - Denial of service (noise, energy injected)
- Hands-off tapping
  - Unshielded copper is an antenna
  - As transmitter (leak) and receiver (interfere)
- Some Layer-2 security

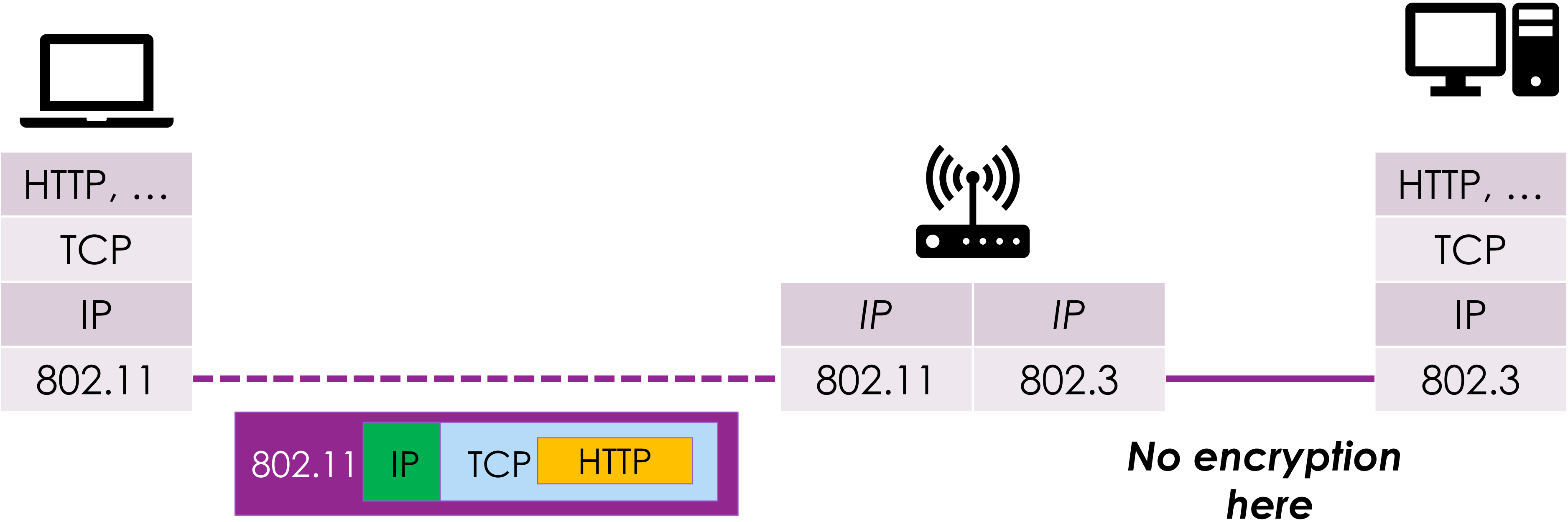
Fibre security

- Also easy to tap
  - Some monitors (oc3mon) used a prism
- Can be easier to detect
  - Energy loss (attenuation?)
  - Quantum entanglement of photons…
- Splicing and cutting (mostly) same as copper
  - Suggestions NSA have done this on the seafloor
- Hands-off tapping
  - Adjacent fibres leak, fibre jackets leak

Wireless security

- By definition, wireless is broadcast
  - Narrow beam antennas help
  - Shorter wavelengths help (optical)
- Take it as read, villains are listening, and can actively intrude
- Each wireless approach is different
  - And limits what it promises
  - Please encrypt at higher layers!
  - But in case you don't…

802.11 – and Wifi Protected Access (WPA1-3)

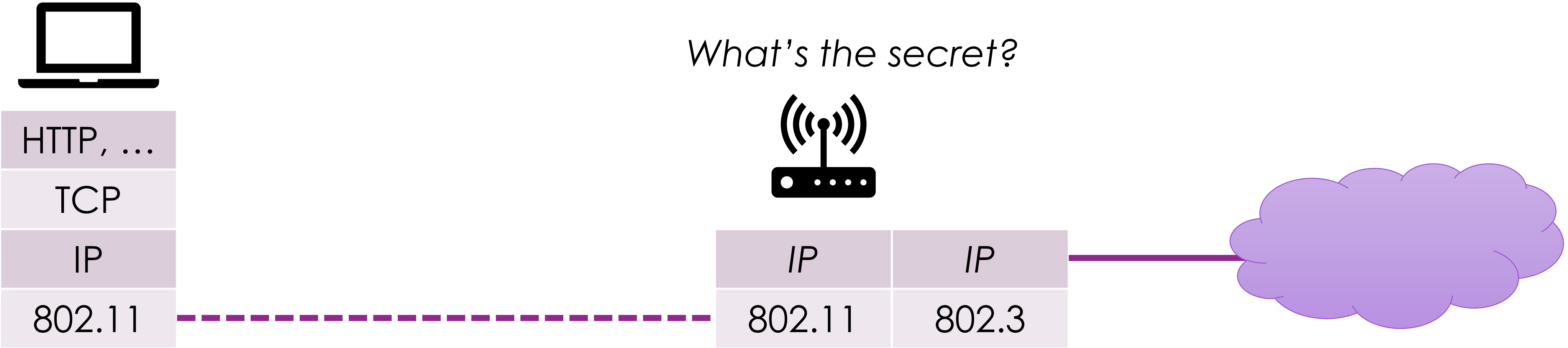


No encryption here

802.11i encrypts frame content

Consider a WiFi Home network

- Typical (un-open) Wifi – pre-shared secret (key) [PSK]
- Client has to prove it (also) knows the (AP) secret
  - Ideally without sending it in plaintext across the network
  - AP then grants access

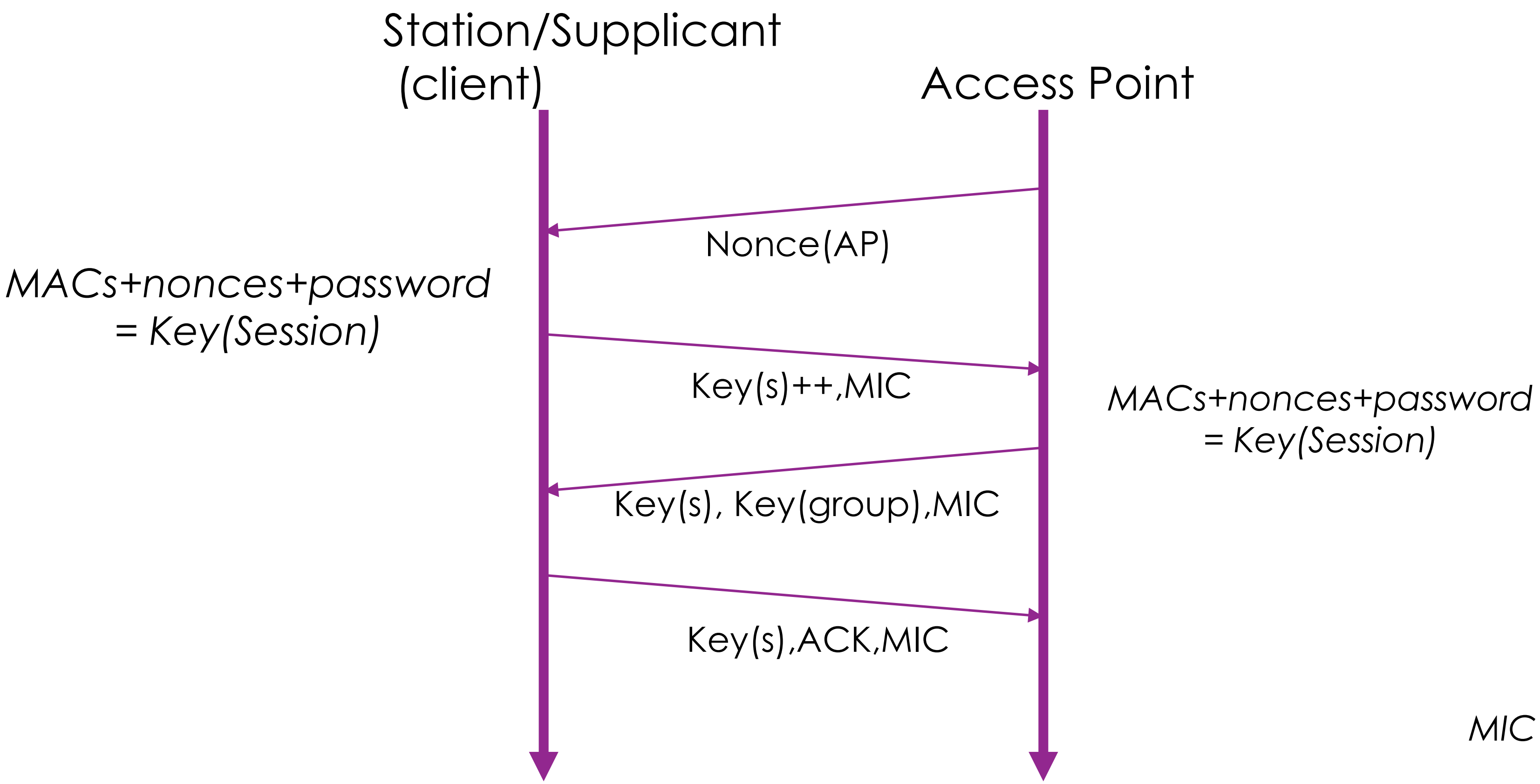


WiFi network keys

HTTP, …
TCP
IP
802.11

- Encryption again - several keys – now at layer 2
- Client authenticates to AP
  - Each calculates a shared session key from the SSID password
    - *Pairwise Transient Key (PTK)*
  - Client tells AP, AP accepts, registers and hands out more keys
- AP-to-clients (broadcast/multicast)
  - Additional group (*temporal*) key *(GTK)*
- Client and AP encrypt with session key
  - Confidentiality, integrity, and authenticity

Keys on keys…



MIC = Message Integrity Code
= a signature

Keys over keys on keys with keys…

- Pairwise Temporal/Transit Key (64 bytes) =
  - Key Confirmation Key (KCK)
  - Key Encryption Key (KEK)
  - Temporal Key (TK)
  - MIC Authenticator (AP) Tx Key
  - MIC Authenticator (AP) Rx Key

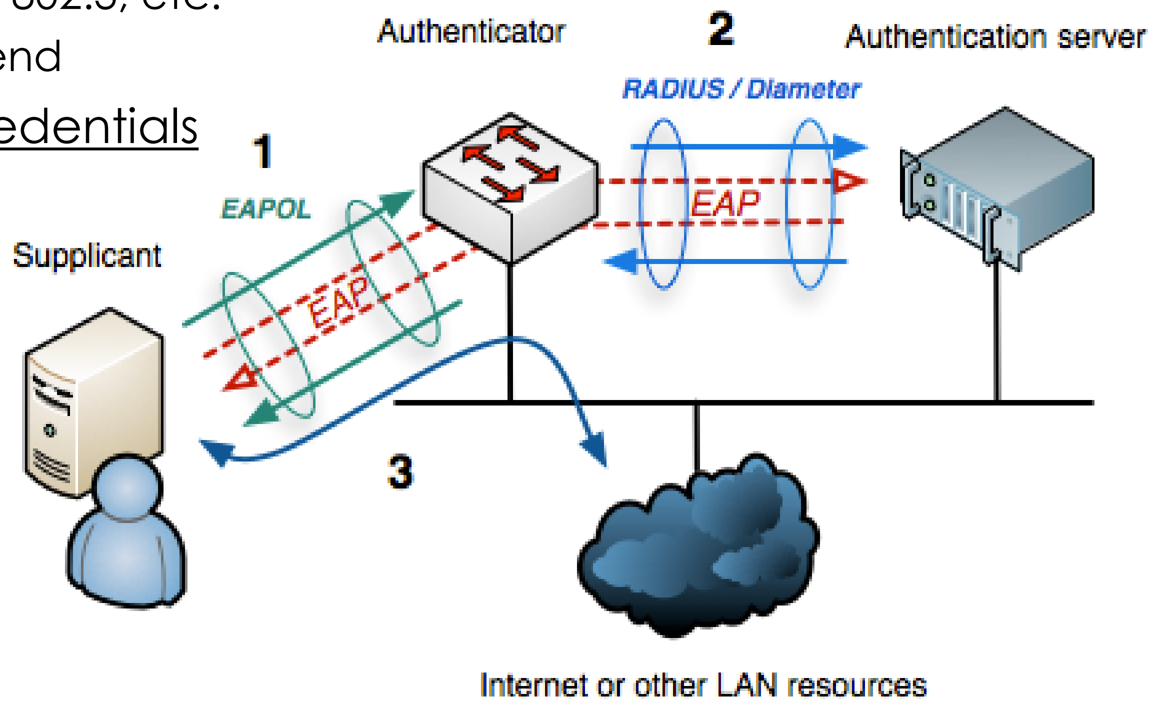
- Group Temporal Key (32 bytes) =
  - Group Temporal Encryption Key
  - MIC Authenticator (AP) Tx Key
  - MIC Authenticator (AP) Rx Key

And more:

- *Pairwise Master Key ~ PSK*
- *Group Master Key*
- *Master Session Key*

Pass-through authentication

- 802.1X
  - Uses *Extensible Authentication Protocol* (EAP)
    - Can be used on 802.11, 802.3, etc.
    - Typically RADIUS back-end
  - <u>Each client has own credentials</u>
    - No PSK



WiFi Encryption history

- Wired Equivalent Privacy WEP
  - Don’t go there. Single key, easily calculated from traffic sniffing.

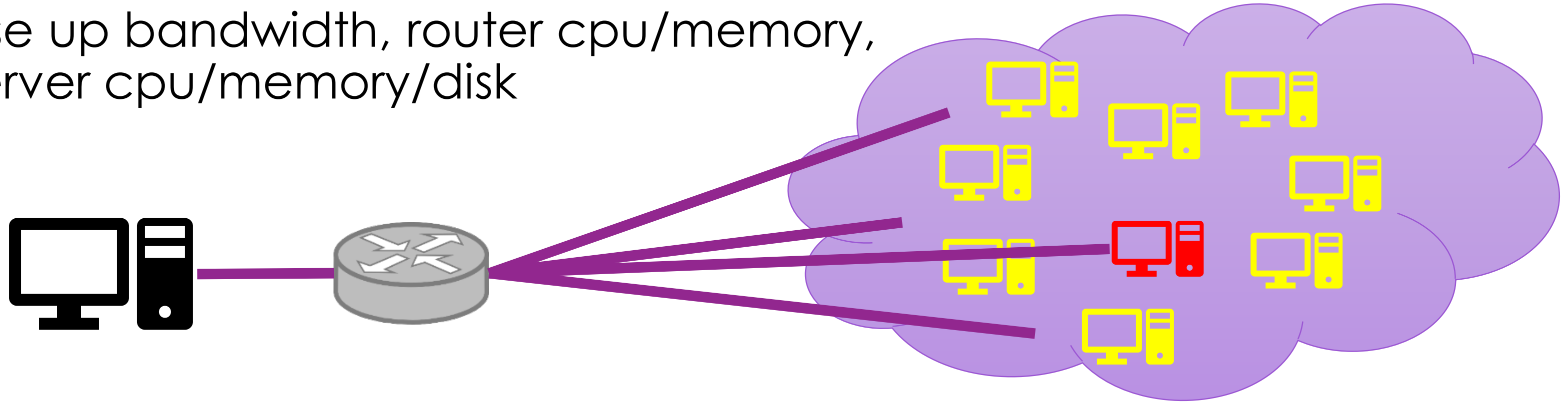
- WiFi Protected Access WPA
  - With Pre-shared-key (PSK)=“personal” or 802.1X=“enterprise”,
    - Better integrity checks than simple CRC
    - Temporal Key Integrity Protocol (TKIP) – per-frame-key
    - Becomes *Counter Mode Cipher Block Chaining Message Authentication Code Protocol (CCMP)*
  - Heaps better. Still broken
    - largely through WPS (“easy-to-join” feature)

- WPA2
  - Lots of additional measures. Much stronger encryption and other protections.
  - Still KRACKed

- WPA3
  - Still warm of the press (Jan 2018)

Other kinds of attack - Denial of Service

- Not all attacks are about eavesdropping or fraud
- Some (many) prevent your systems from being available to intended users
  - "Take down" your website, booking system, banking portal, government site, …
  - For fun or fortune!
- Based on resource starvation
  - Encryption can't help you now!
  - Use up bandwidth, router cpu/memory, server cpu/memory/disk



DoS vectors

- Tricky packets at different layers…
- Ping of Death
  - Large ICMP packet, multiple fragments, modified headers
  - Reassembled into >64kB – memory overflow
  - Actually a direct attack on the target host
- SYN flood
  - Open a TCP connection to a server
  - But never follow through on SYN/ACK
  - Server builds connection state for each inbound packet
  - Can be avoided with SYN Cookies
    - Server builds connection state only after ACK

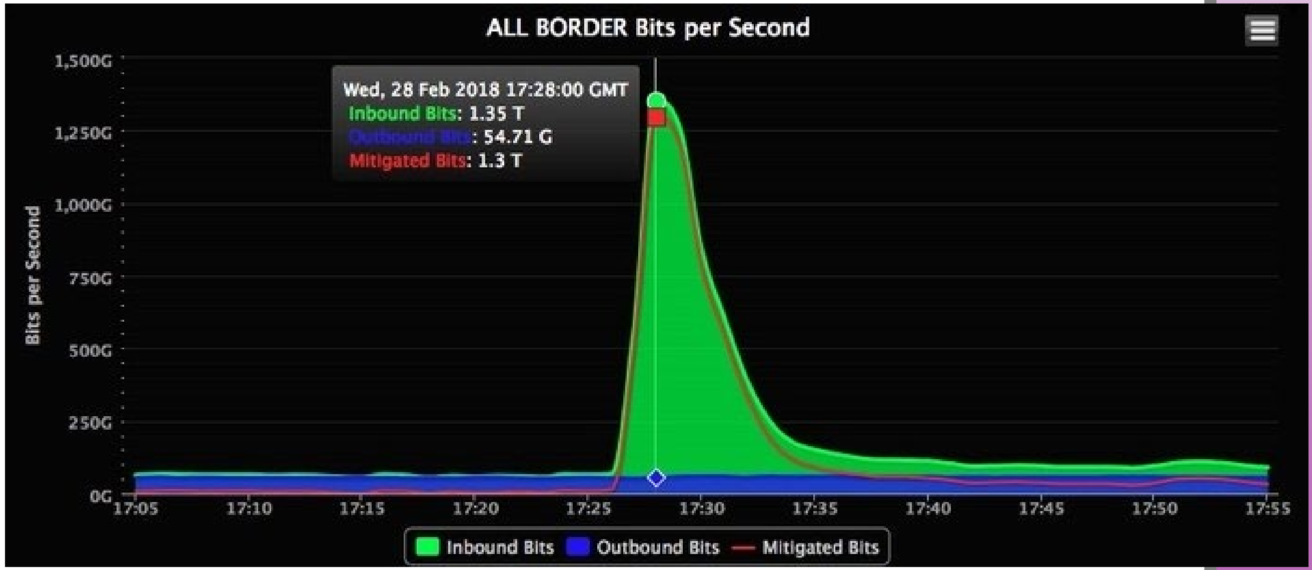
DoS vectors

- Application layer messages:
- Making small but complex queries
  - Big database queries
  - Complex regex
- Memory overflows
- Other bugs

- Can starve server

Distributed Denial of Service

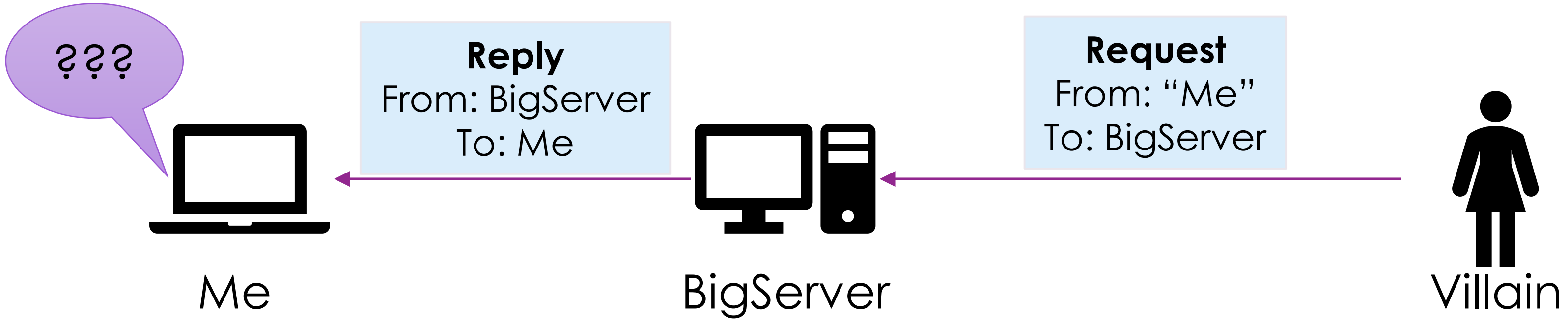
- Could flood somebody with a single server
- But hey, it's the internet, let's use **millions**
  - IoT devices, webcams, NVR, baby monitors, smartphones, … "botnets"
  - More common than desktops, laptops
- Scale?
  - 10Million attacks/year, at 1-2Gb/s
  - 2016 – first 1Terabit/s DDoS attack
    - *100,000+ wireless webcams*
  - 2017 – multiple 1Tb/s attacks



2018.02.28 - github

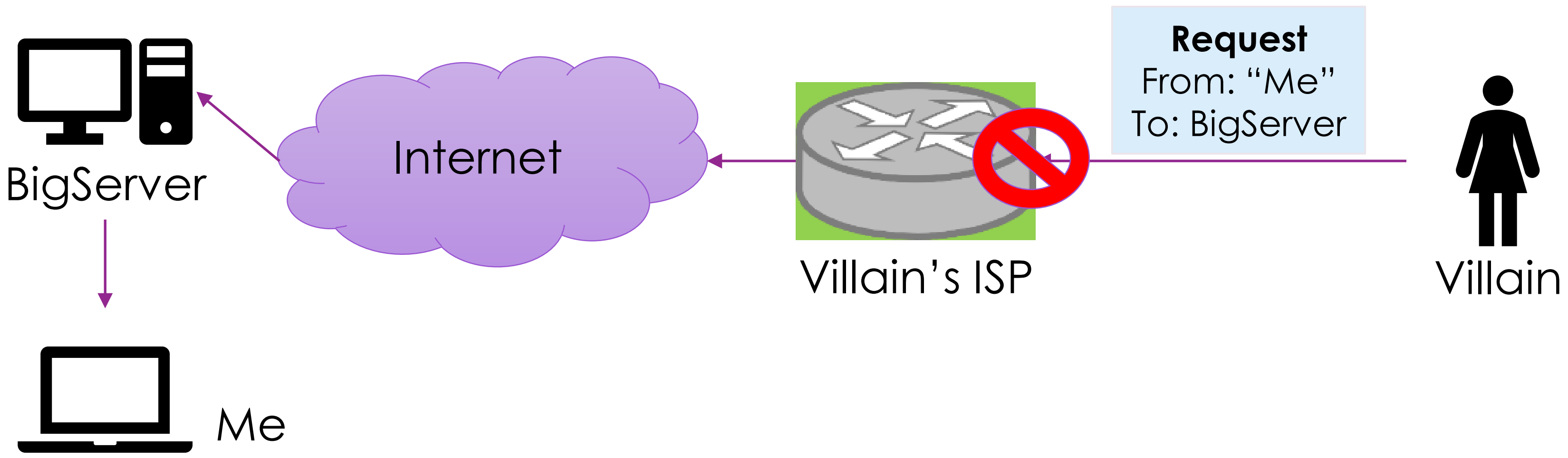
“Spoofing”

- Many DoS attacks spoof
- Sending packets with false source addresses
  - Get other boxes to work for me
  - Very hard to trace



Spoofing vs Ingress Filtering

- Good practice:
  - Check Source IP at <u>their</u> boundary to the Internet
    - Nobody else can…
  - Obvious? Obvious!
  - And yet not widely enabled (more work, and only benefits others)



Multipliers

- **Host multipliers**: Botnets…
  - Lots of hacked devices, controlled centrally
  - Each sends any old kind of packet (e.g. ping flood, udp, …)

- **Packet multipliers** – send one, get many – often with ‘spoofing’
- SMURF (and variations)
  - Ping the broadcast address (one packet out)
  - Use the target’s IP address as source (spoofing)
  - Everyone replies to the source (many packets back)
  - Easy to prevent…

Packet multiplication

- Small requests, big responses
- DNS
  - 1 packet request "list the hosts inside anu.edu.au"
  - Multi-megabyte response
- HTTP/FTP/NFS/…
  - 1 packet request "Send me that 10GB file"
  - …10GB later…?
- Memcache servers
  - Database accelerators – over UDP
  - Should be only inside your network – and yet there are thousands visible

Mitigation – really hard

- The Internet is designed to shift lots of packets…

- Content Distribution Networks
  - Don’t be a single target

- Edge routers/Attack detection
  - Efficient packet dropping
  - Better filtering support (e.g. DoS from a particular country?)

- Upstream provider support
  - Can re-route most/all traffic elsewhere
  - E.g. dedicated DDoS processing systems

- Get ingress filtering everywhere!
  - And fix all those webcams while you’re at it…

Are we done?

- Network security and design: will never be done!
- But, for us, yeah, we're done.
- Just 4 more guest lectures!
- And a final word from our sponsors…

Australian National University

The Student Experience of Learning &Teaching (SELT) is changing in 2019

NEW SELT SURVEY

ANU listened to your feedback => new course and teacher surveys

- Clearer, more focused questions, so that it is easier to complete the survey.
- Fewer questions, so that completing the survey will take much less time.

Ensured some elements remained unchanged:

- You can still provide feedback through other channels.
- Responses are anonymous.
- Responses are voluntary but highly encouraged.
- Survey results will be released to staff after grades are released.