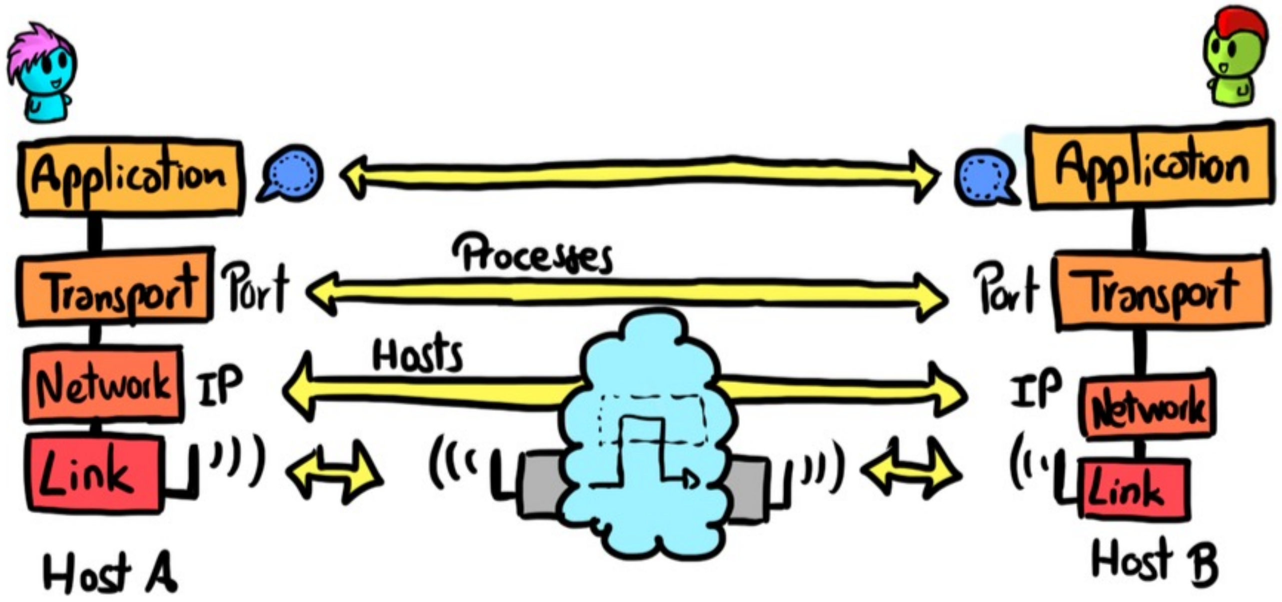
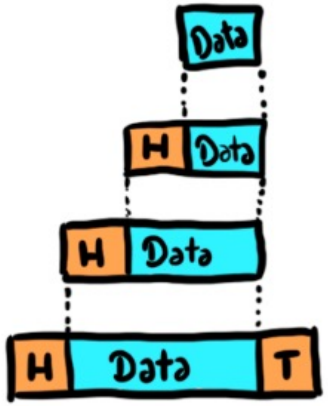
CS489/698 Privacy, Cryptography, Network and Data Security

Security through the layers

Spring 2024, Monday/Wednesday 11:30am-12:50pm

Recall, the Network Stack



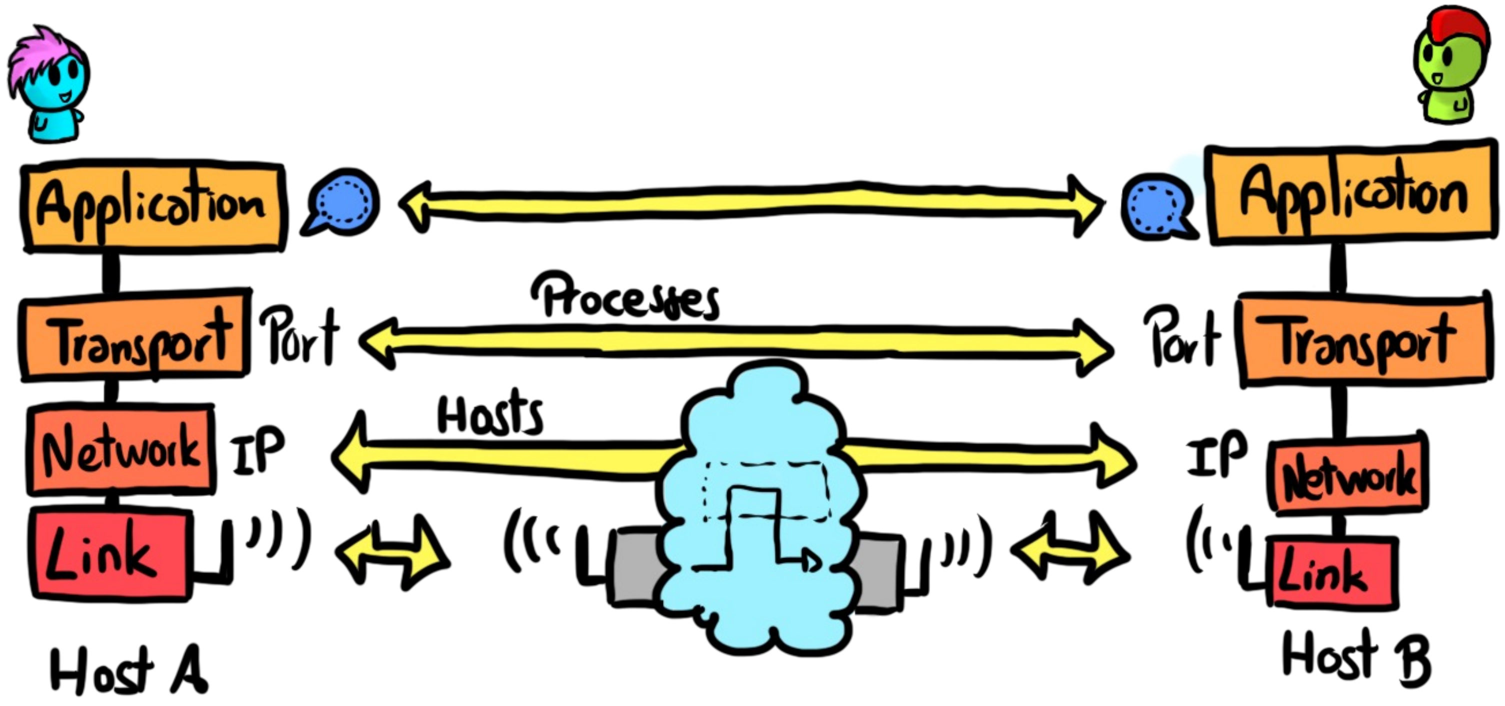


Q: Where do we need to apply crypto? (confidentiality, integrity, authentication)

- A Link layer is enough
- B Application layer is enough
- C We need it in all layers
- D Who needs crypto?

Today's Lecture – Security through the layers

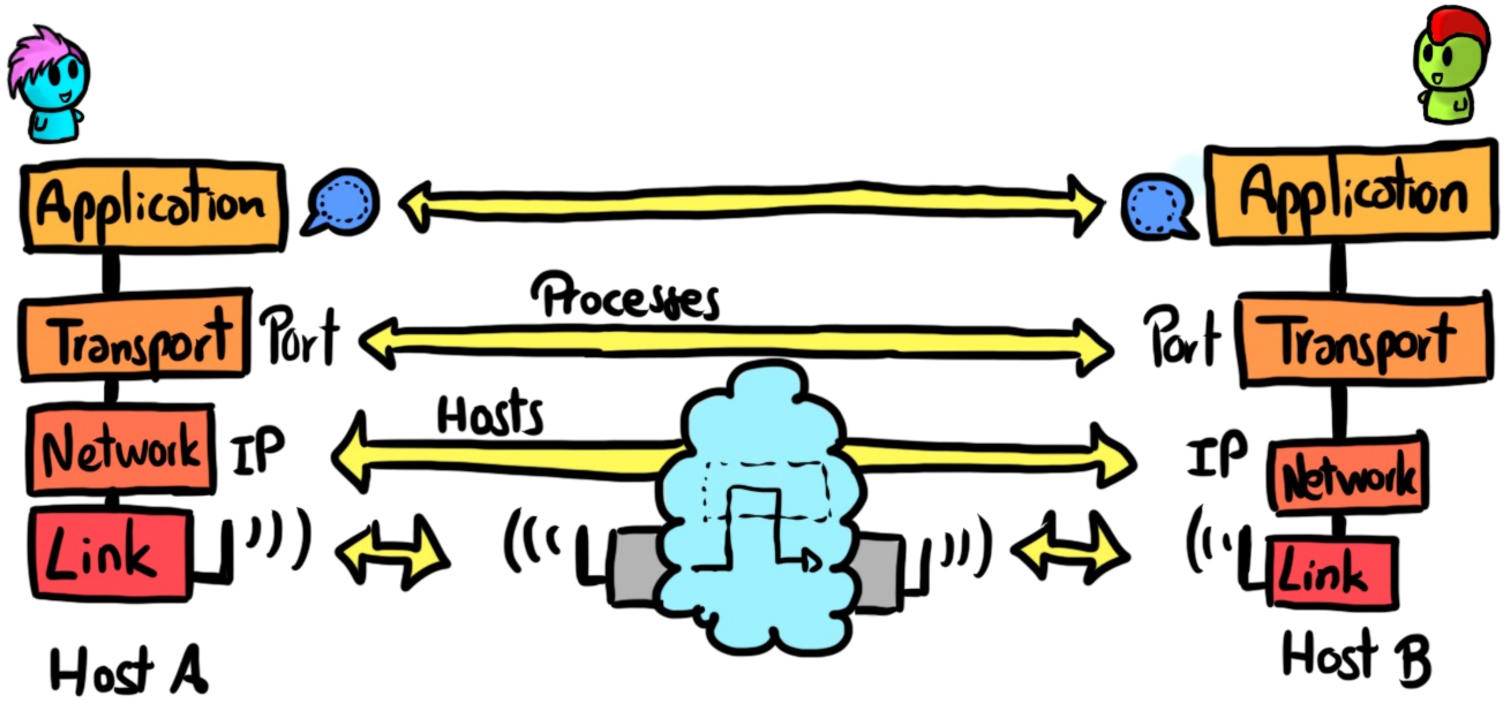
- Link
  - WEP, WPA, WPA2
- Network
  - VPN, IPsec
- Transport
  - TLS/SSL
- Application
  - ssh, (Next class: PGP, OTR, Signal)



Link Layer – WPA2

Security through the layers

- Link
  - WEP, WPA, WPA2
- Network
  - VPN, IPsec
- Transport
  - TLS/SSL
- Application
  - ssh, (Next class: PGP, OTR, Signal)



The history of Wi-Fi Security

- WEP - Learn From Mistakes
  - 1999
- WPA - Temporary Patch
  - 2003
- WPA2 - Mostly Ok
  - 2004
- WPA3 – Current Standard
  - 2018

Wired Equivalent Privacy (WEP)

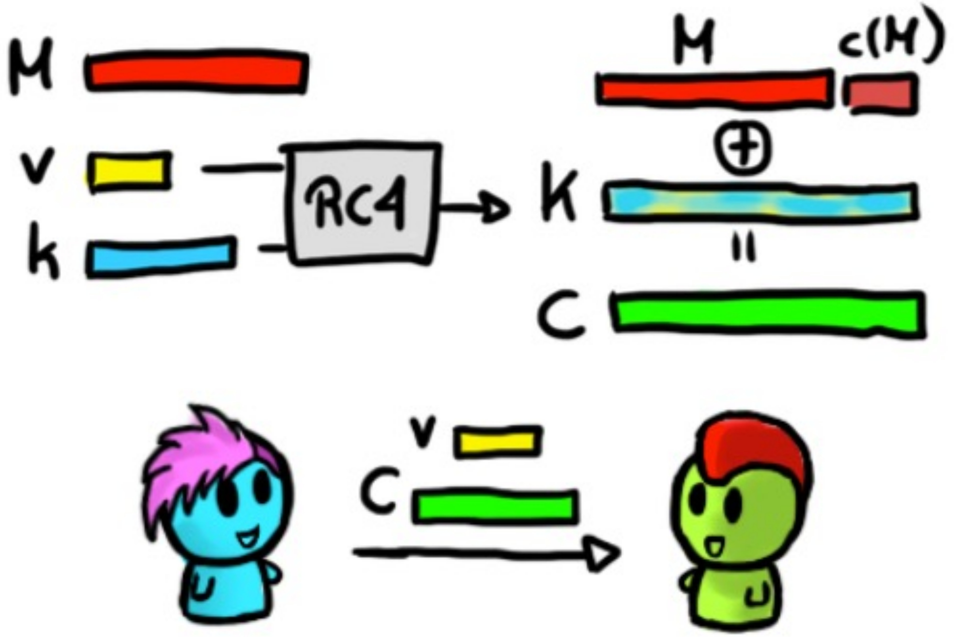
WEP was intended to enforce three security goals:

- Data Confidentiality
  - Prevent an adversary from learning the contents of the wireless traffic
- Data Integrity
  - Prevent an adversary from modifying the wireless traffic or fabricating traffic that looks legitimate
- Access Control
  - Prevent an adversary from using your wireless infrastructure

Unfortunately, none of these are actually enforced!

WEP Protocol

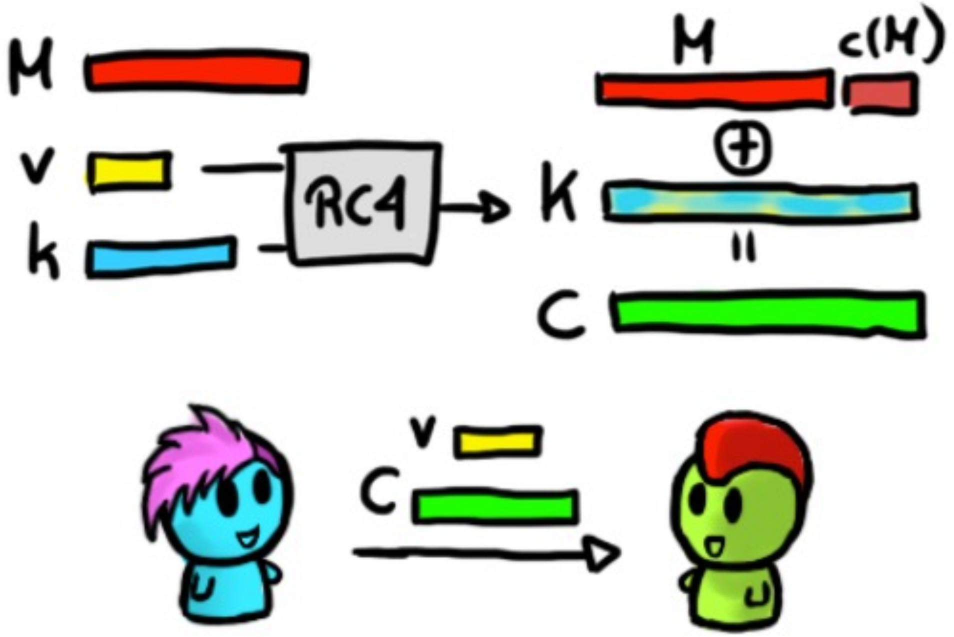
- The sender and receiver share a secret k (either 40 or 104 bits)
- In order to transmit a message M:
  - Compute a checksum $c(M)$ (which does not depend on k)
  - Pick an IV v and generate a keystream $K = RC4(v, k)$
  - Ciphertext $C = K \oplus \langle M \parallel c(M) \rangle$
  - Transmit v and C over the wireless link



Q: What kind of cipher is this?

WEP Protocol

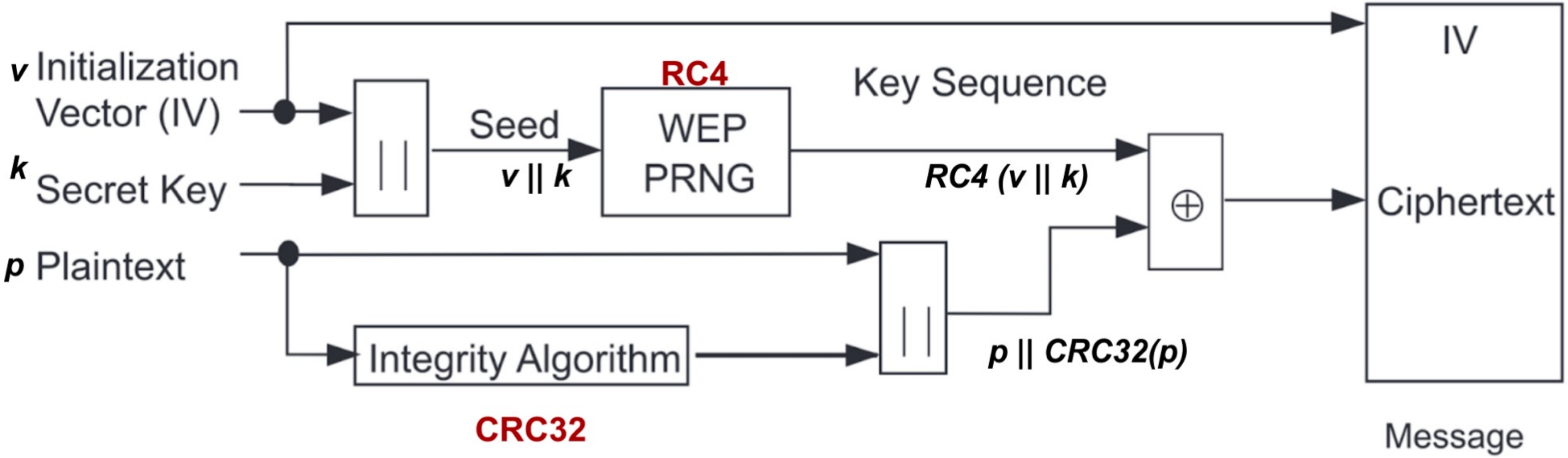
- The sender and receiver share a secret k (either 40 or 104 bits)
- In order to transmit a message M:
  - Compute a checksum $c(M)$ (which does not depend on k)
  - Pick an IV v and generate a keystream $K = RC4(v, k)$
  - Ciphertext $C = K \oplus \langle M \parallel c(M) \rangle$
  - Transmit v and C over the wireless link



Q: What kind of cipher is this?

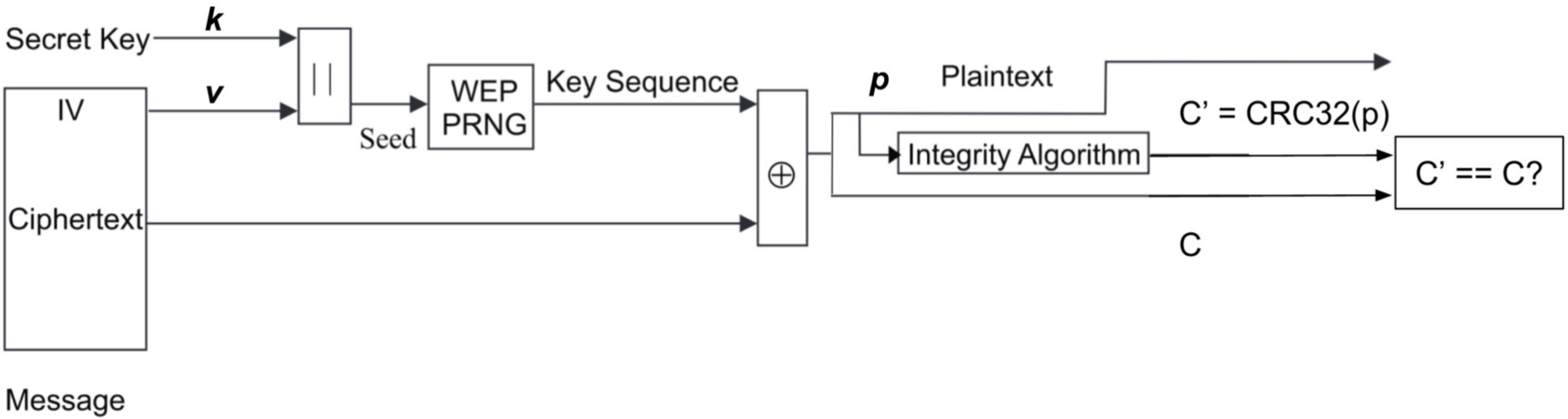
A: It's a stream cipher (symmetric)

WEP Encryption Algorithm

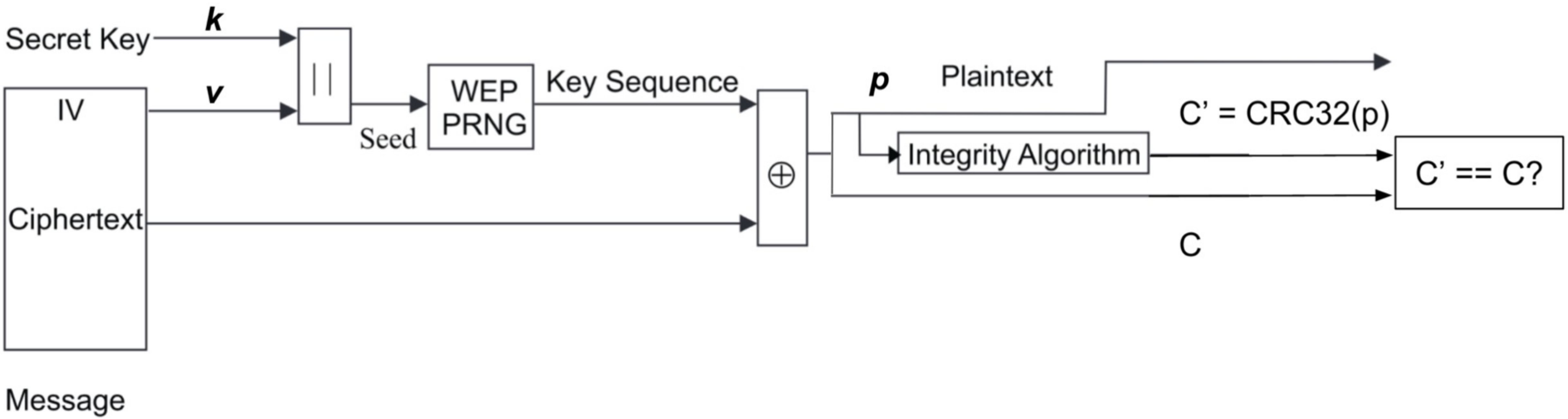


Q: How do we decrypt?

WEP Decryption

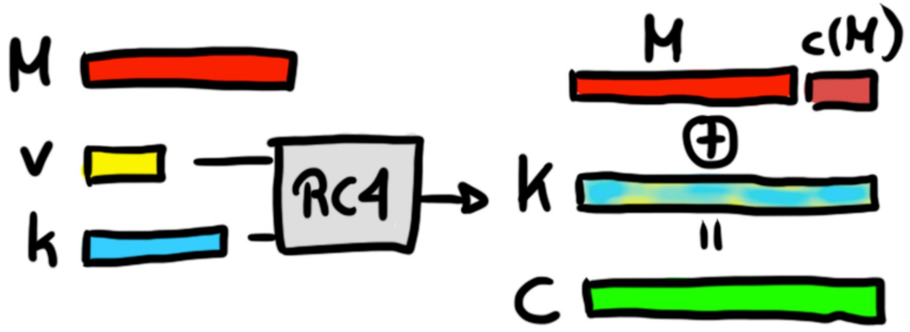


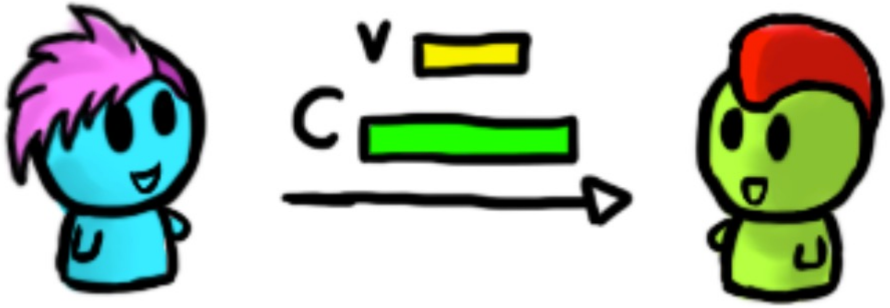
WEP Decryption



Looks… ok? What's the issue?

Problem 1: Key Reuse

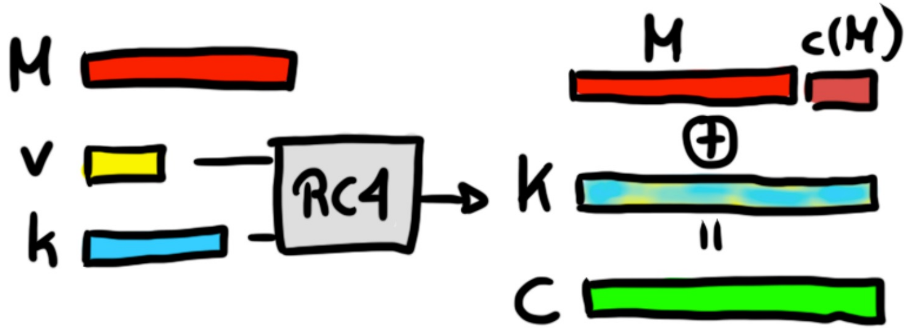


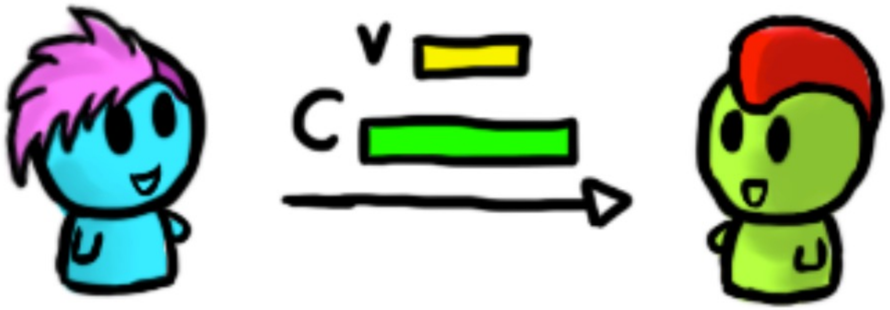


- IV (v) is too short: only 3 bytes = 24 bits.
- Secret (k) is rarely changed!

Q: What is the problem with this?

Problem 1: Key Reuse



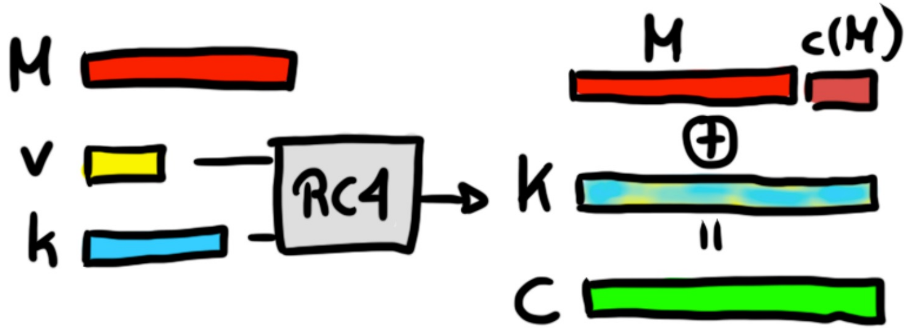


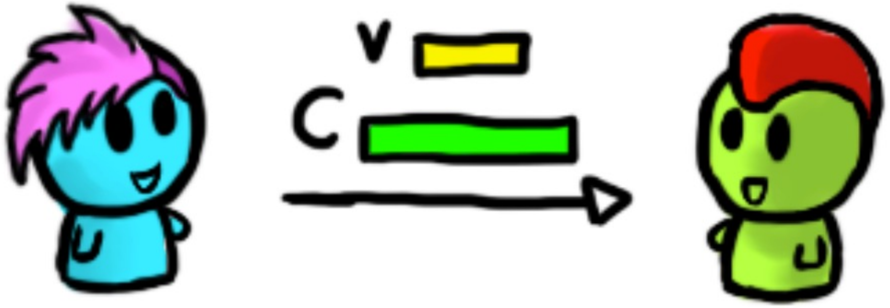
- IV (v) is too short: only 3 bytes = 24 bits.
- Secret (k) is rarely changed!

Q: What is the problem with this?

A: Key-stream gets re-used after 2^{24} iterations (~17M packets) → two-time pad

Problem 2: Integrity?





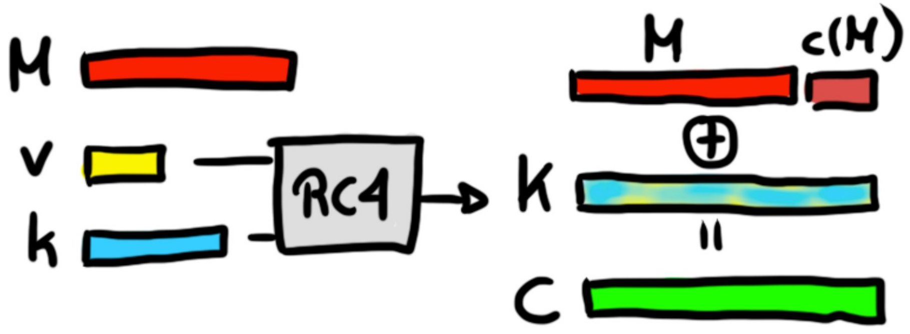
The checksum algorithm in WEP is CRC32, which has two <u>undesirable</u> properties:

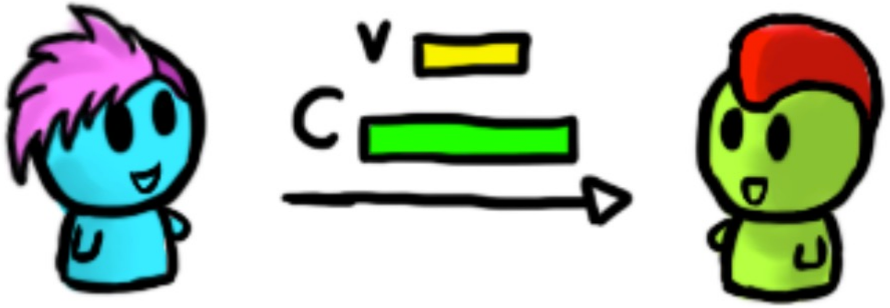
- It is independent of k and v
- It is linear: $c(M \oplus \delta) = c(M) \oplus c(\delta)$

Q: What is the problem with this?

Problem 2: Integrity?

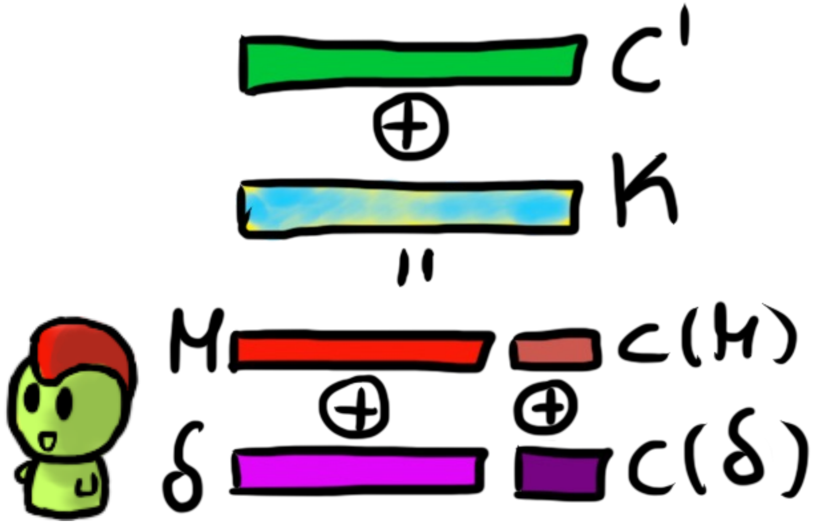
A: See below





The sender transmits C and v. If Mallory wants to modify the plaintext M into M' = M ⊕ δ:

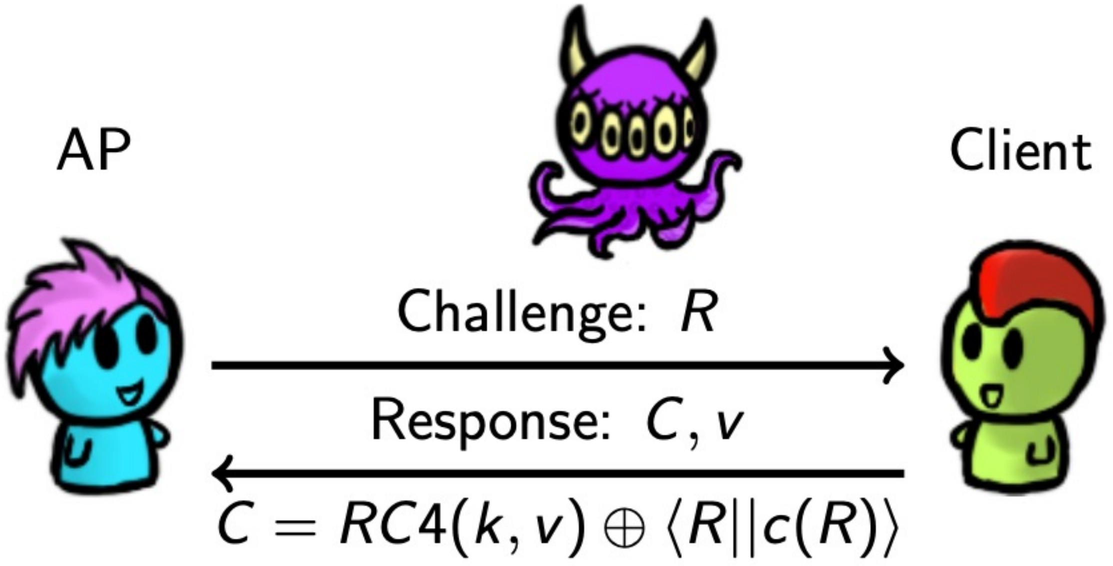
- Calculate C' = ⟨M || c(M)⟩ ⊕ ⟨δ || c(δ)⟩
- Send (C',v) instead of (C,v)
- This passes the integrity check of the recipient!



Q: How can we avoid this?

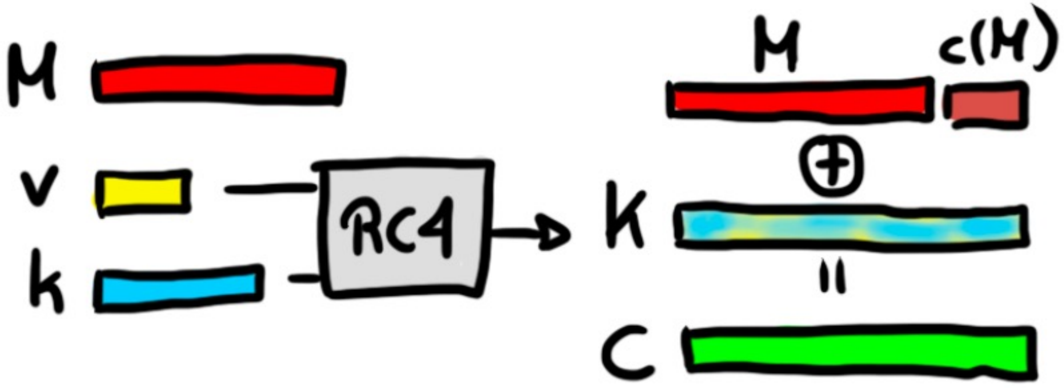
How does WEP Authenticate?

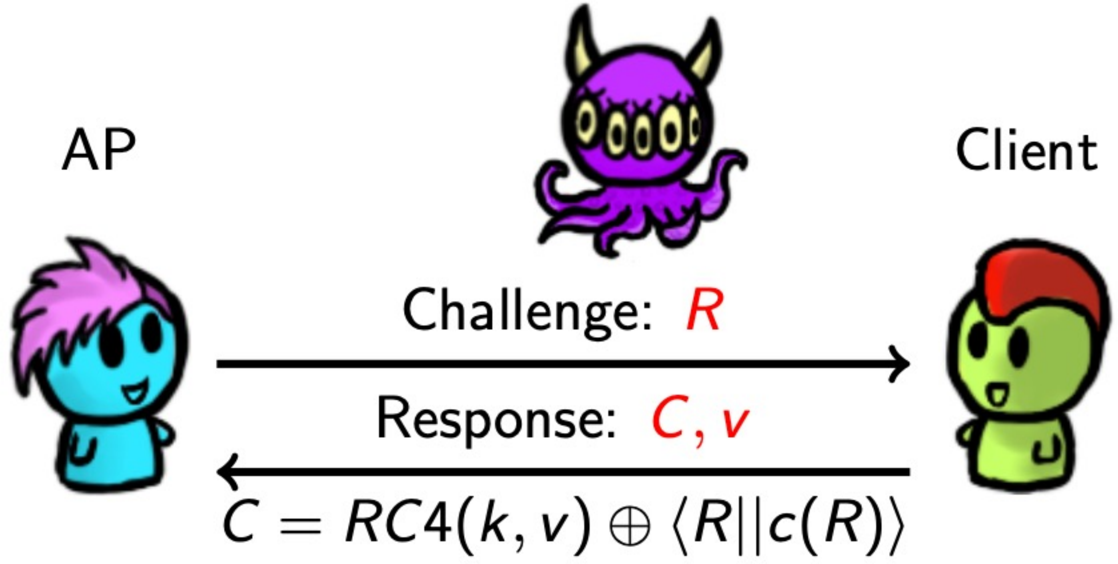
- **R** is a random challenge string
- Client encrypts **R** to prove knowledge of **k** to the AP
- If encrypted correctly, AP accepts client!



AP = Access Point, a.k.a. your wireless router

Let’s break WEP authentication!

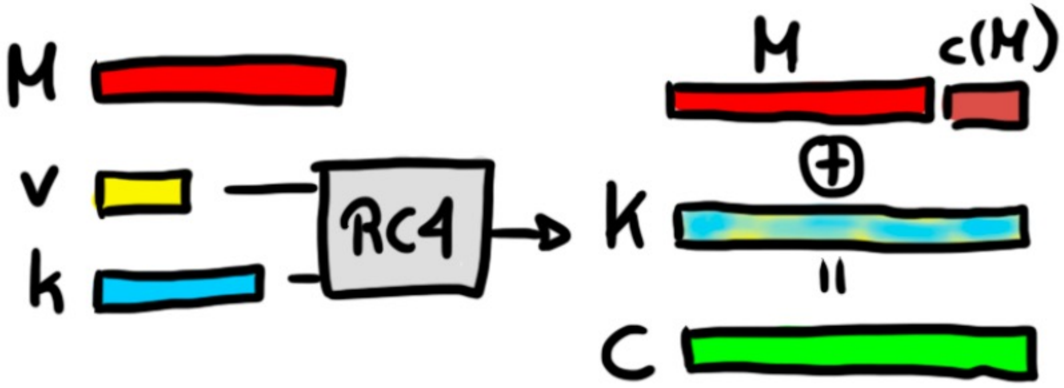


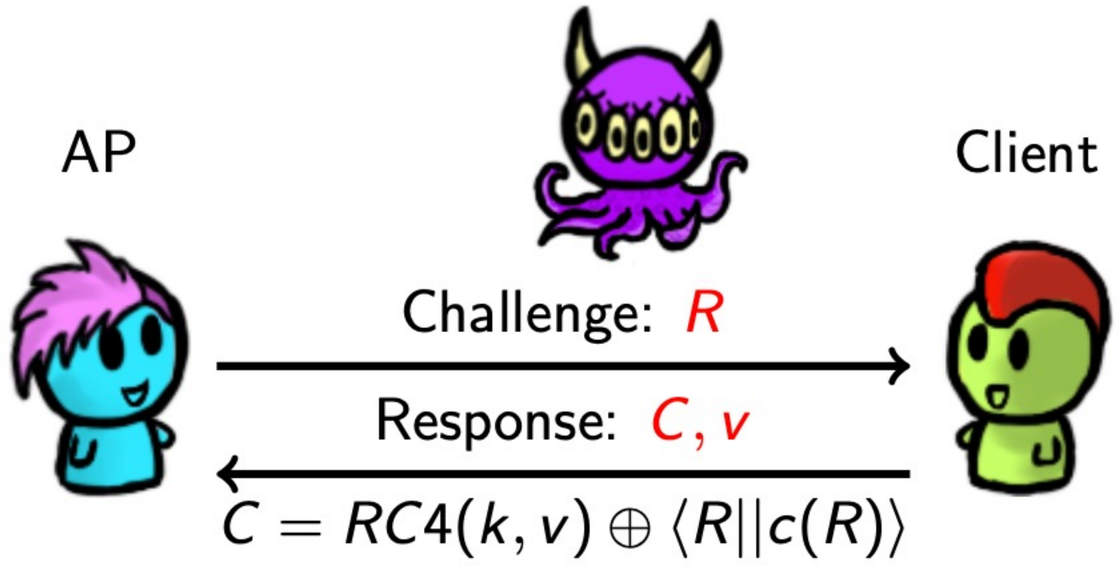


Mallory has seen R, C, and v.

Q: Mallory wants to authenticate herself to the AP. The AP sends Mallory a new challenge R′. Can Mallory successfully run the authentication protocol?

Let's break WEP authentication!





Mallory has seen R, C, and v.

Q: Mallory wants to authenticate herself to the AP. The AP sends Mallory a new challenge R′. Can Mallory successfully run the authentication protocol?

A: Yes! Note that Mallory knows RC4(k,v) = C ⨁ ⟨R||c(R)⟩. Mallory can just compute: C′ = RC4(k,v) ⨁ ⟨R′||c(R′)⟩ and send C′ and v to the AP.

Problem 3: Packet injection!?!

- We saw that seeing **R**, **C**, and **v** gives Mallory a value of **v** and the corresponding keystream **RC4(v,k)**
- The same way Mallory encrypted the challenge R' in the previous slide, she can encrypt any other value **F**:
  - $C' = \langle F \| c(F) \rangle \oplus RC4(v,k)$, and she transmits v,C'
- C' is a correct encryption of F, so the AP accepts the message

Problem 3: Packet injection!?!

- We saw that seeing **R**, **C**, and **v** gives Mallory a value of **v** and the corresponding keystream **RC4(v,k)**
- The same way Mallory encrypted the challenge R' in the previous slide, she can encrypt any other value **F**:
  - $C' = \langle F \| c(F) \rangle \oplus RC4(v,k)$, and she transmits v,C'
- C' is a correct encryption of F, so the AP accepts the message

So what?

Escalate

- Somewhat surprisingly, the ability to modify and inject packets leads to ways in which Mallory can trick the AP to **decrypt** packets!
  - Check out Prof. Goldberg's talk if you are interested.

- None of the attacks so far use the fact that the stream cipher was RC4
  - When RC4 is used with similar keys, the output keystream has a subtle weakness
  - Leads to recovery of either a 104-bit or 40-bit WEP key in under 60 seconds
  - Check this paper for more details

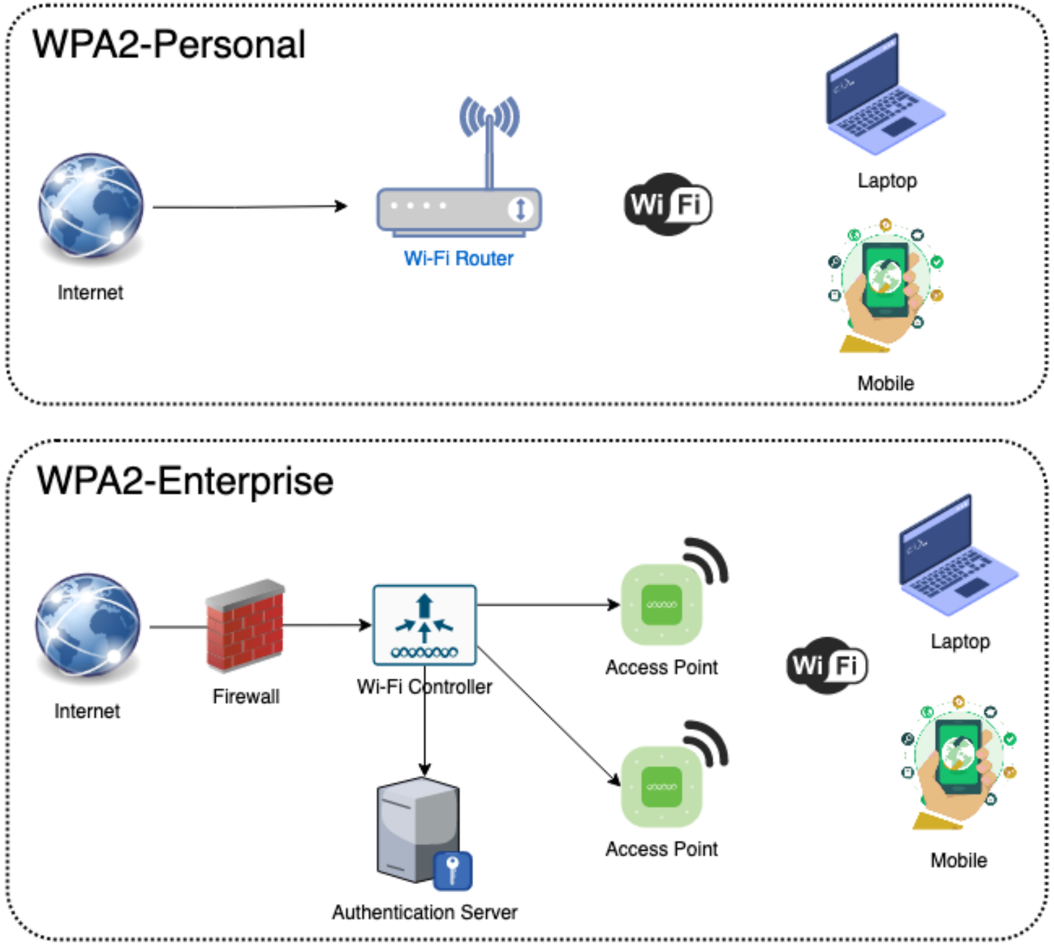
What have we learnt from WEP?

- Have sufficient randomness
  - Use long keys and long IVs.
  - Don’t reuse short-term secret keys and IVs.
- Do not use checksums for integrity. Use keyed MACs instead! They are not linear.
- Go through public reviews of cryptographic protocols before standardizing them! This helps find weaknesses.

Wi-Fi Protected Access II (WPA2)

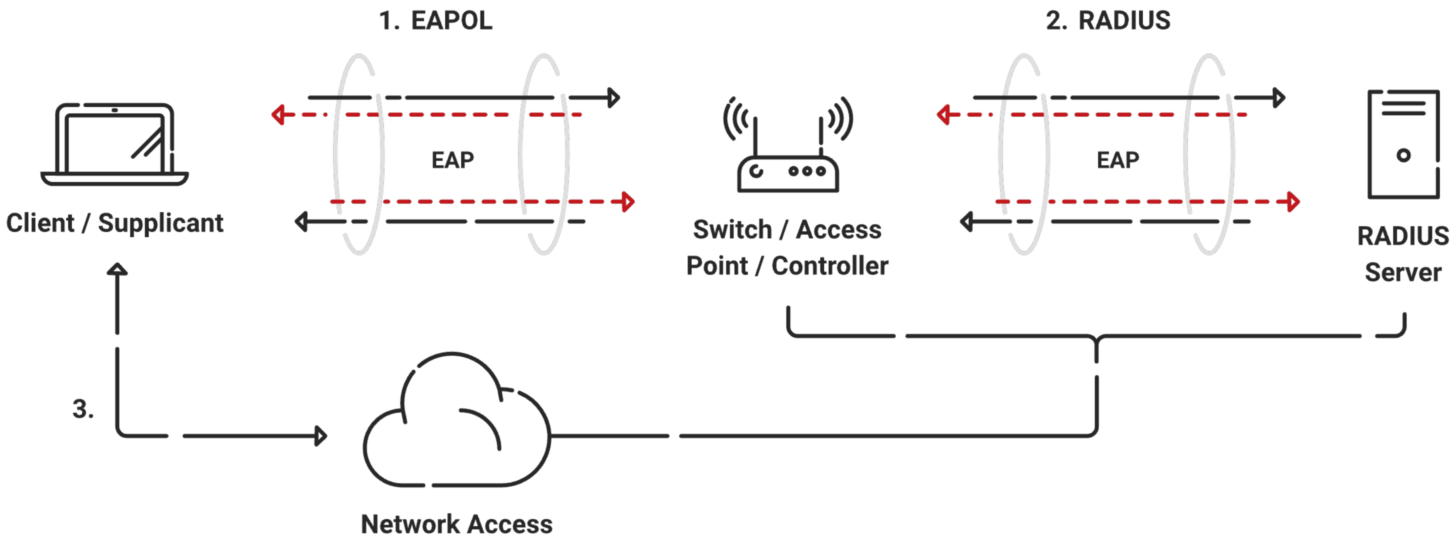
- Has been required for products calling themselves “Wi-fi” since 2006
- Replaces RC4 with the CCM authenticated encryption mode (using AES)
- IV is 48 bit
- Replaces checksum with a real MAC
- Key is changed frequently
- Ability to use a 802.1x authentication server
    - But maintains a less-secure PSK (Pre-Shared Key) mode for home users

Two Types - Very different protocols



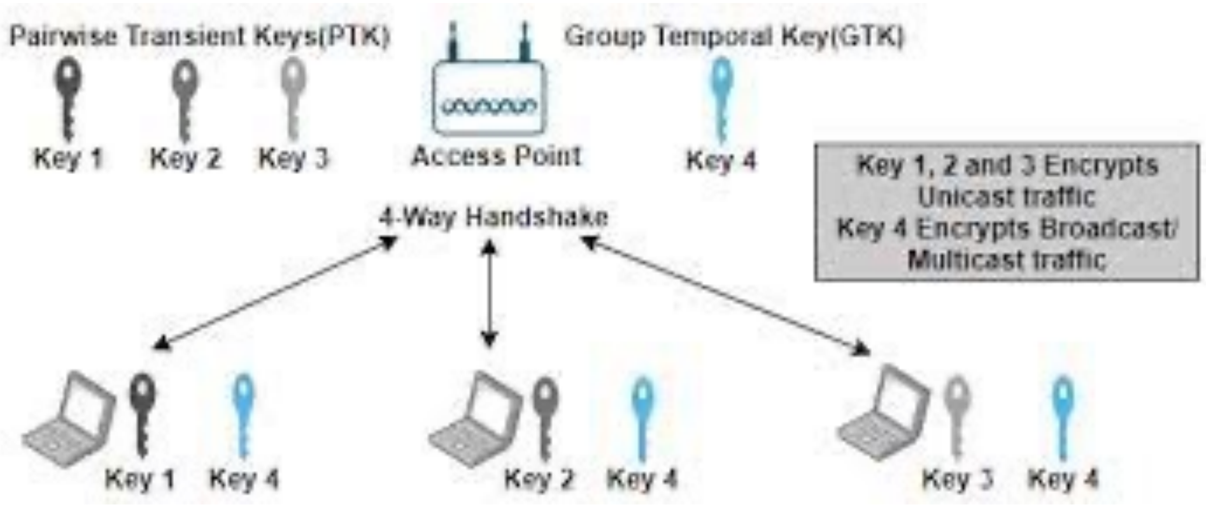
Step 1 - Authentication

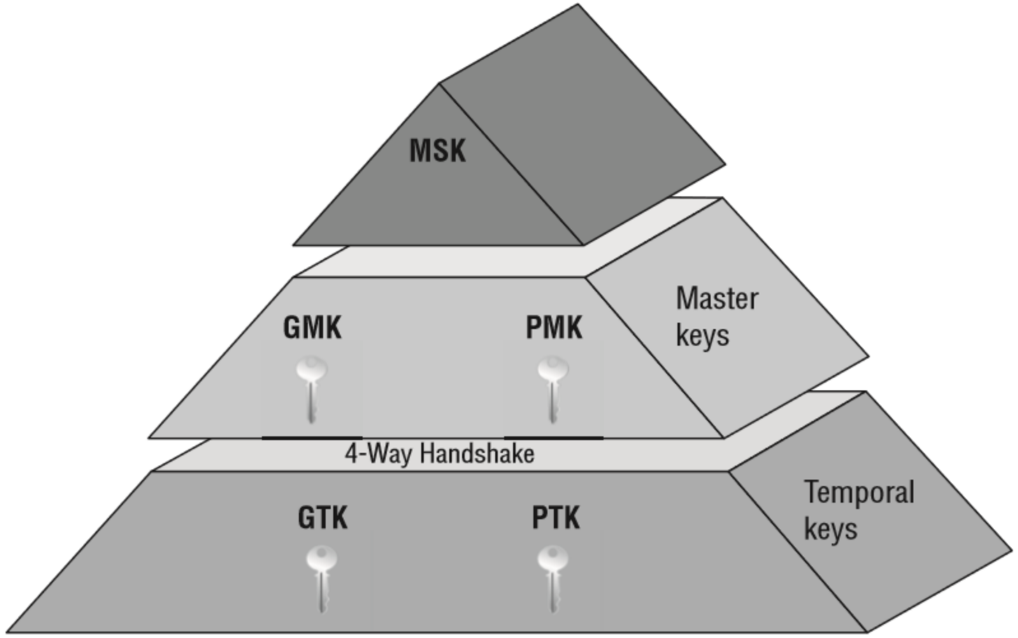
- In enterprise mode, run EAP protocol to obtain PMK



- In personal mode, PMK= pre-existing PSK (like a password)

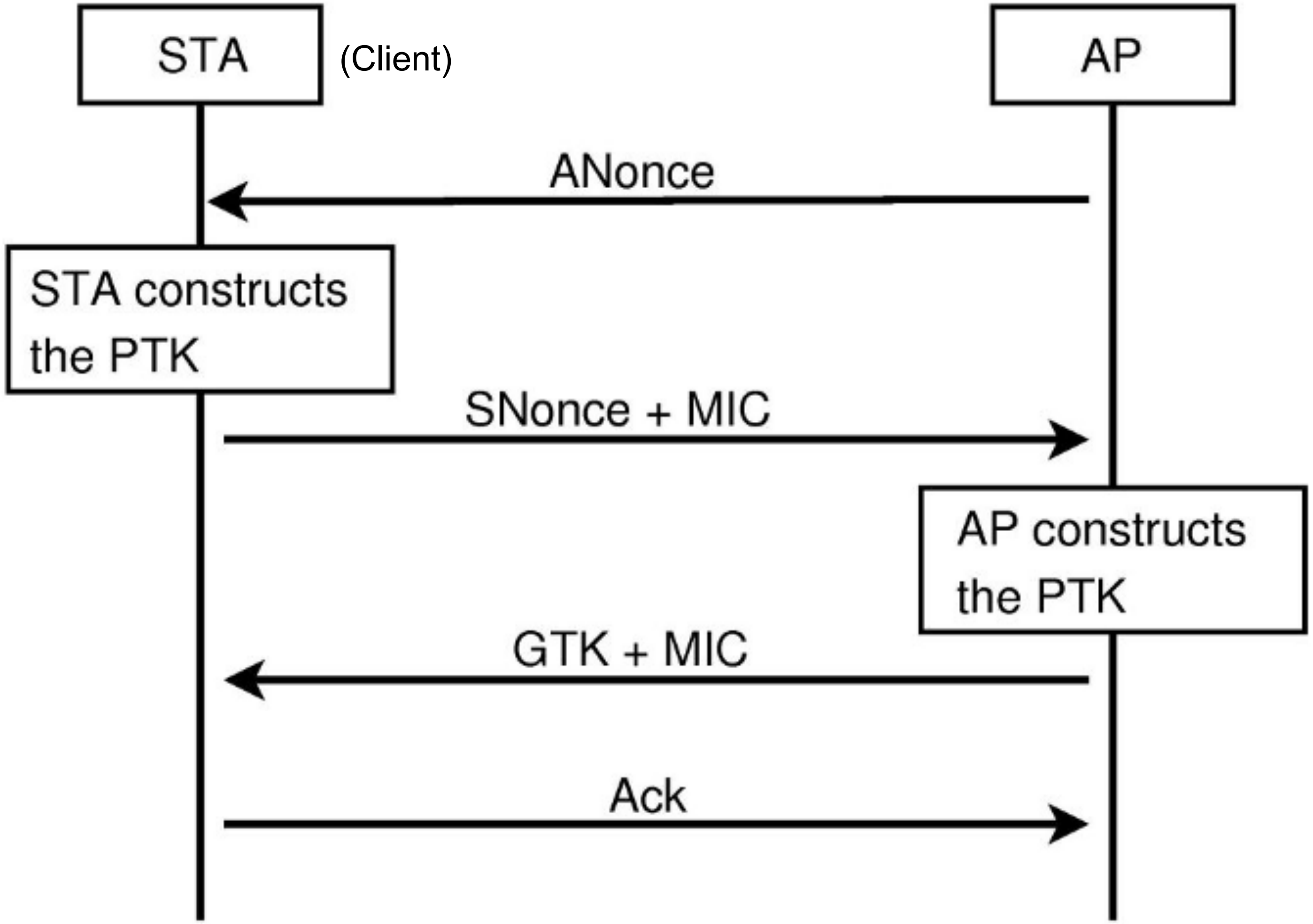
Step 2 - Deriving the Keys





4 Way Handshake

- PTK = PMK + ANONCE + SNONCE + MAC(AA) + MAC(SA)



Attacks Against WPA2

- Offline dictionary attack:
  - Only in the personal version
  - 4-way handshake is in plaintext: attacker observes all parts of PTK except PSK/PMK
  - Also see MIC. Can guess and check until MIC is valid.
- KRACK attack:
  - Resending message 3 causes an old key to be used!
  - Depending on cipher, two-time pad attack works
  - Patched in WPA2 and 3

Wi-Fi Protected Access III (WPA3)

- Uses dragonfly key exchange to prevent offline dictionary attacks
- Uses stronger crypto protocols
  - Only AES and Better Hash
- So far so good…

Comparing Wi-Fi Protocols

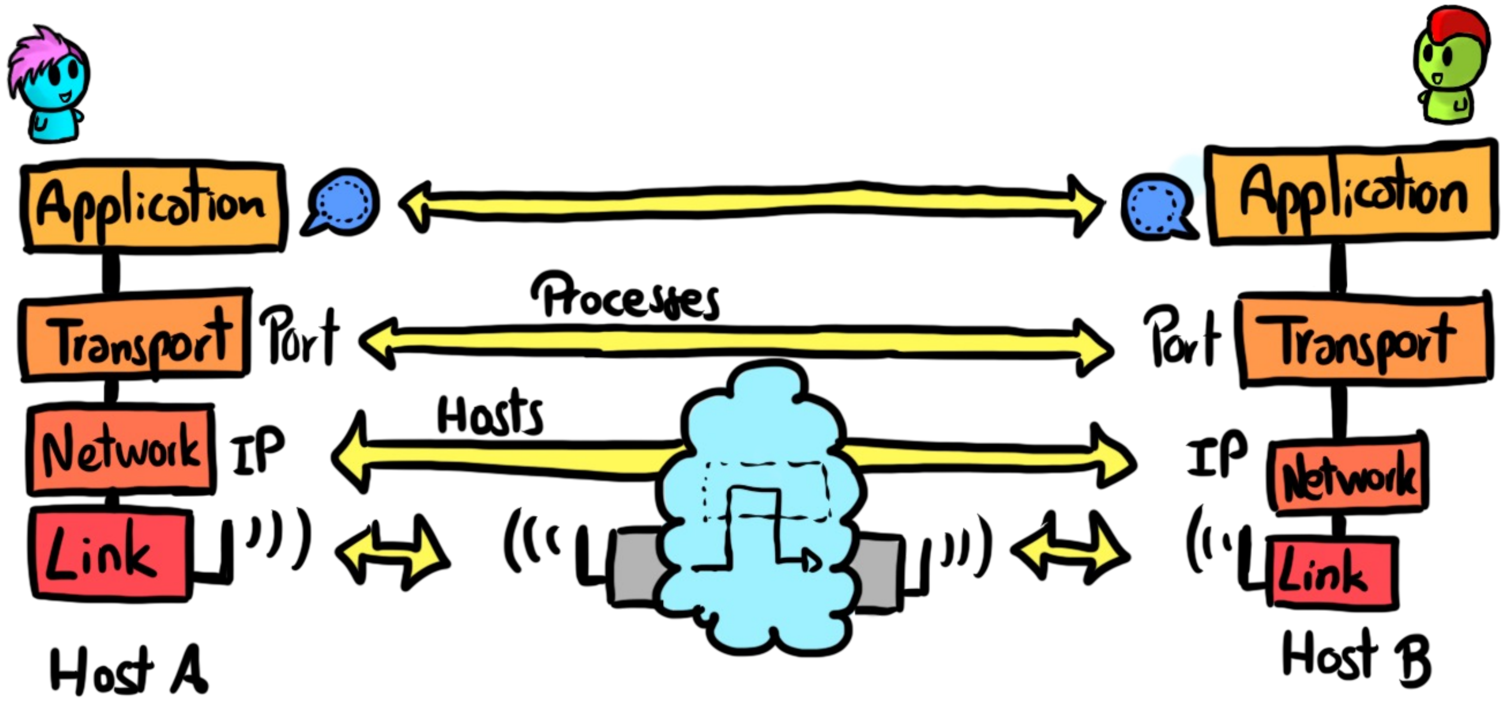
	WEP	WPA	WPA2	WPA3
Release Year	1999	2003	2004	2018
Encryption Method	Rivest Clipher 4 (RC4)	Temporal Key Integrity Protocol(TKIP) with RC4	CCMP and Advanced Encryption Standard	Advanced Encryption Standard(AES)
Session Key Size	40-bit	128-bit	128-bit	128-bit(WPA3-Personal) 192-bit(WPA3-Enterprise)
Clipher Type	Stream	Stream	Block	Block
Data Integrity	CRC-32	Message Integrity Code	CBC-MAC	Secure Hash Algorithm
Key Management	Not provided	4-way handshaking mechanism	4-way handshaking mechanism	Simultaneous Authentication of Equals handshark
Authentication	WPE-Open WPE-Shared	Pre-Shared Key(PSK)& 802.1x with EAP variant	Pre-Shared Key(PSK)& 802.1x with EAP variant	Simultaneous Authentication of Equals(SAE)&802.1x with EAP variant

Source: FS Community

Network Layer – VPNs

Security through the layers

- Link
  - WEP, WPA, WPA2
- Network
  - VPN, IPsec
- Transport
  - TLS/SSL
- Application
  - ssh, (Next class: PGP, OTR, Signal)



Why do we need network layer security?

Suppose every link in our network had strong link-layer security. Why would this not be enough?

- Source & destination IPs may not share the same link.
    - Prone to network layer threats such as IP spoofing.
- We need end-to-end security across networks.

IP Security suite (IPSec)

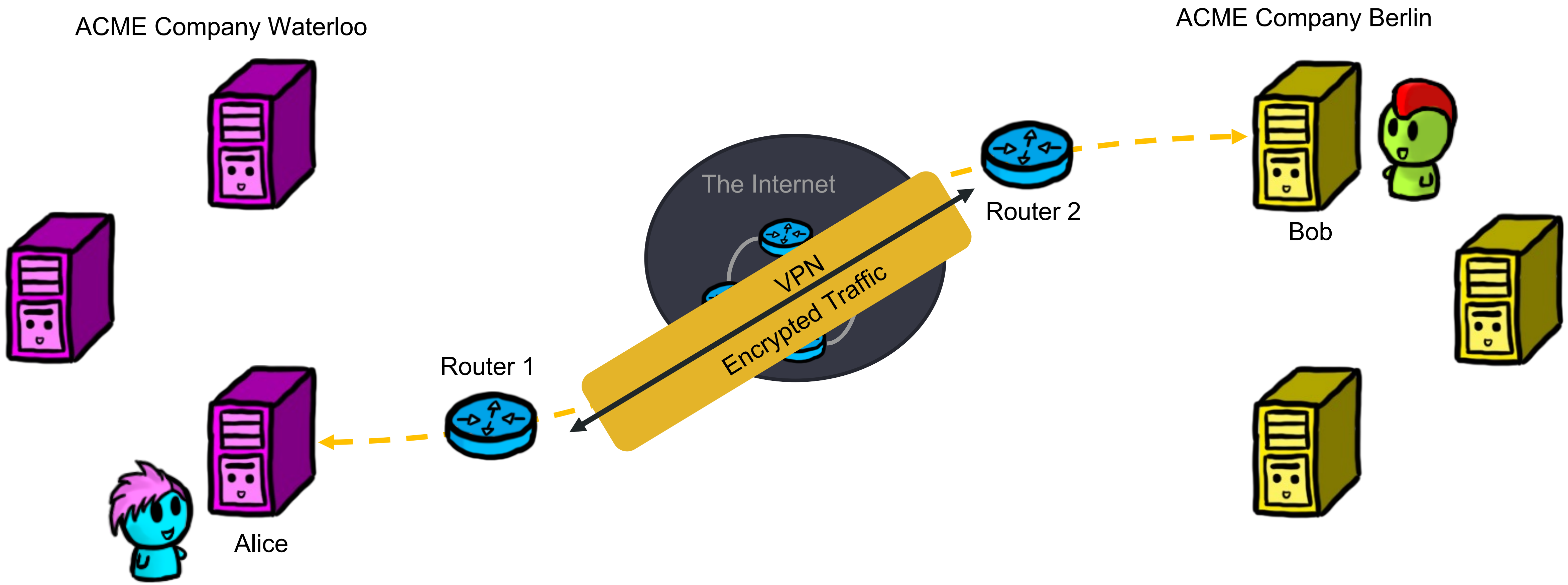
- Extends IP to provide confidentiality and integrity.
- It has two main modes:
  - Transport Mode
  - Tunnel Mode

Virtual Private Networks

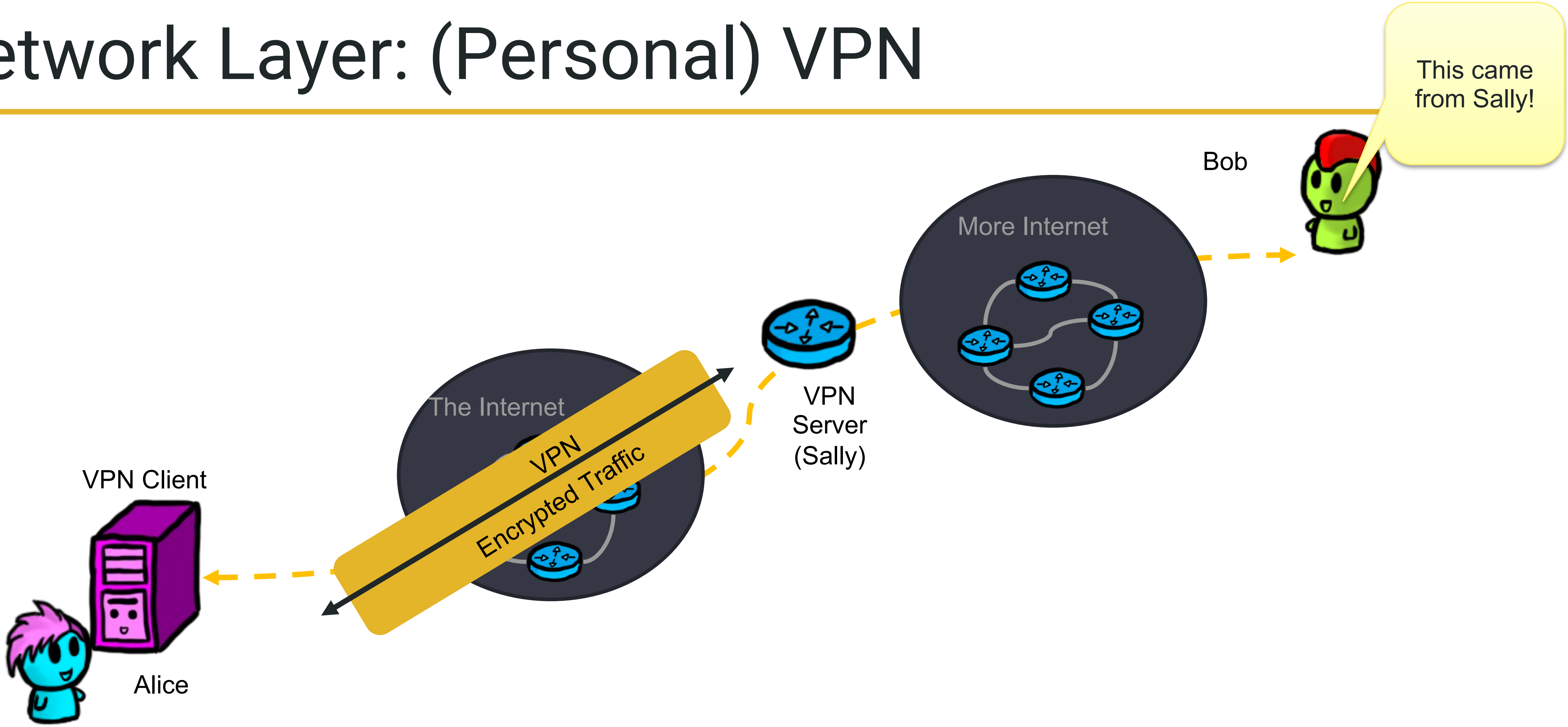
Private network: Has firewalls, access control and authentication so it is only used by trusted users.

Virtual private network: A private network that connects physically distant users via virtual links, that are secured via cryptography

Network Layer: (Corporate) VPN



Network Layer: (Personal) VPN

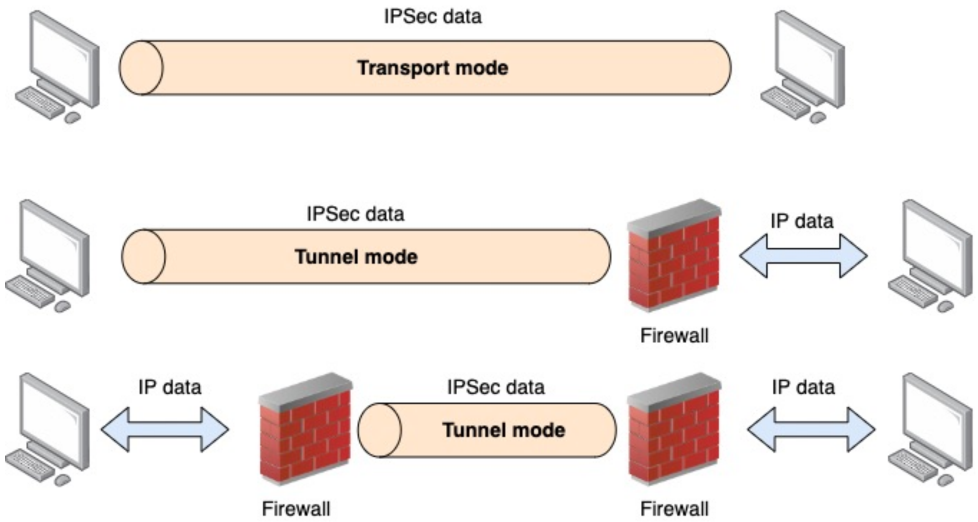


“Interesting” Traffic

- In a corporate VPN, the VPN gateway can be configured to protect only “interesting traffic”
  - Furthermore, different types of traffic can go down different tunnels
- Similar to a firewall
  - Can be based on IP address, type of traffic, etc.
- Not usually the case in personal VPNs that protect everything.
- Once again false positives and negatives are important

Transport Vs Tunnel Mode

- Transport: for point-to-point protection
  - Does not hide the IPs
  - Just encrypts the IP payload
  - Less common
- Tunnel: network-to-network or point-to-network
  - Hides the IP header and payload
  - Multiple variants:
    - Extend network across internet
    - Working remotely
    - Personal VPNs



Components of IPSec

1. Security Association (**SA**): Determine algorithms and keys
    - Decide MAC and encryption scheme (typically AES and SHA256), generate keys etc.
    - Uses IKE protocol.
2. Then, add either:

    A) Authentication Header (**AH**):
    - Provides integrity only

    B) Encapsulating Security Protocol (**ESP**)
    - Provides integrity and encryption

- How do we distinguish between the two IPSec formats?
    - "Protocol" field in the first IP header is set to 50 for AH, 51 for ESP

IKE V2 – At a high level

Two tunnels:

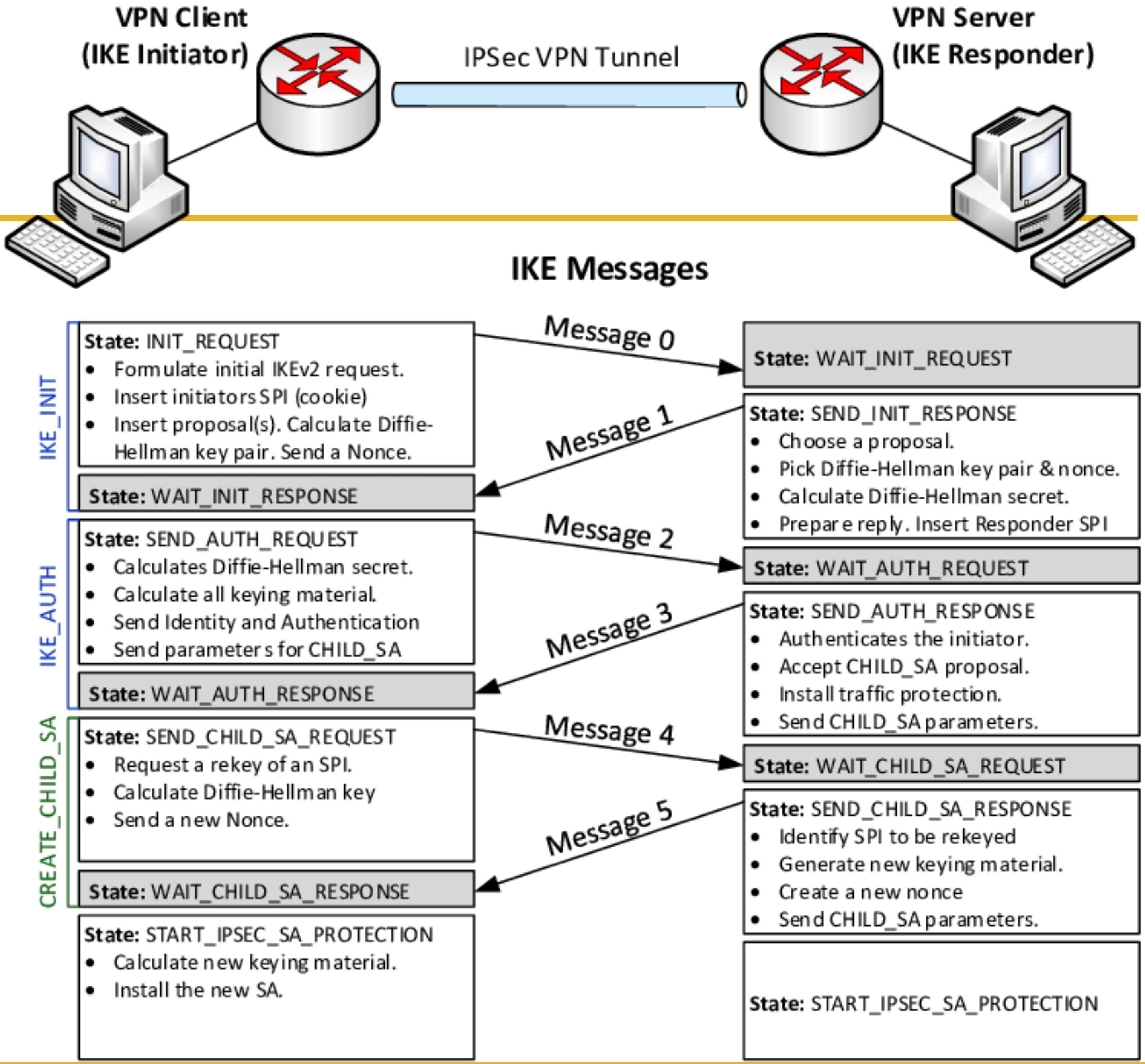
- Initial longer term parent tunnel (first DH)
- Short term child tunnel renewed more often (second DH)

Basic Procedure:

1) Agree on protocols etc.
2) Do DH and authenticate to create master SA
3) Create child SA (including AH vs ESP and new keys)

IKE V2 – The details

- First 4 messages create first tunnel
- Last 4 messages create the second tunnel…
- We won't worry about the details



Source: Hakeem et al.

Authentication Header (AH)

- Offers integrity and data source authentication
    - Authenticates payload and parts of IP header that do not get modified during transfer, e.g., source IP address
- Offers protection against replay attacks
    - Via extended sequence numbers

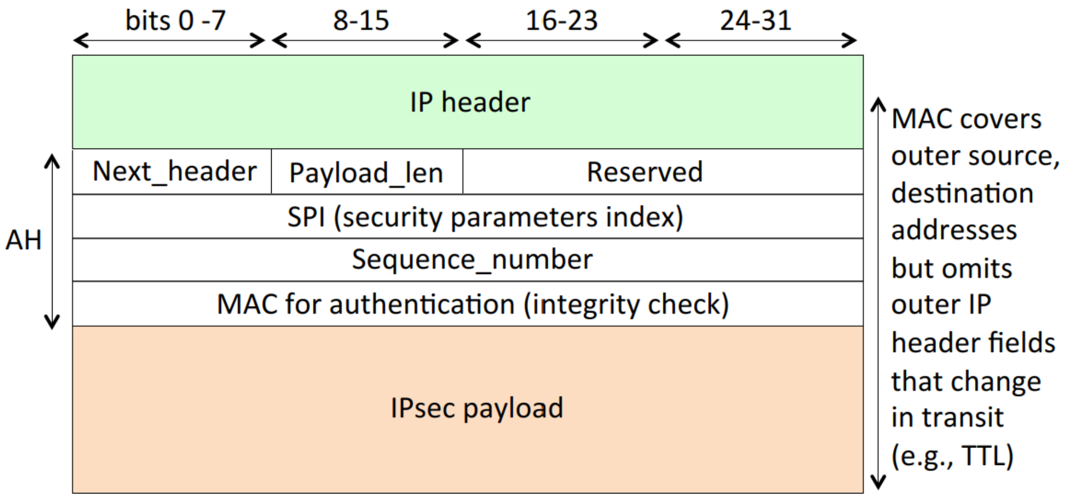
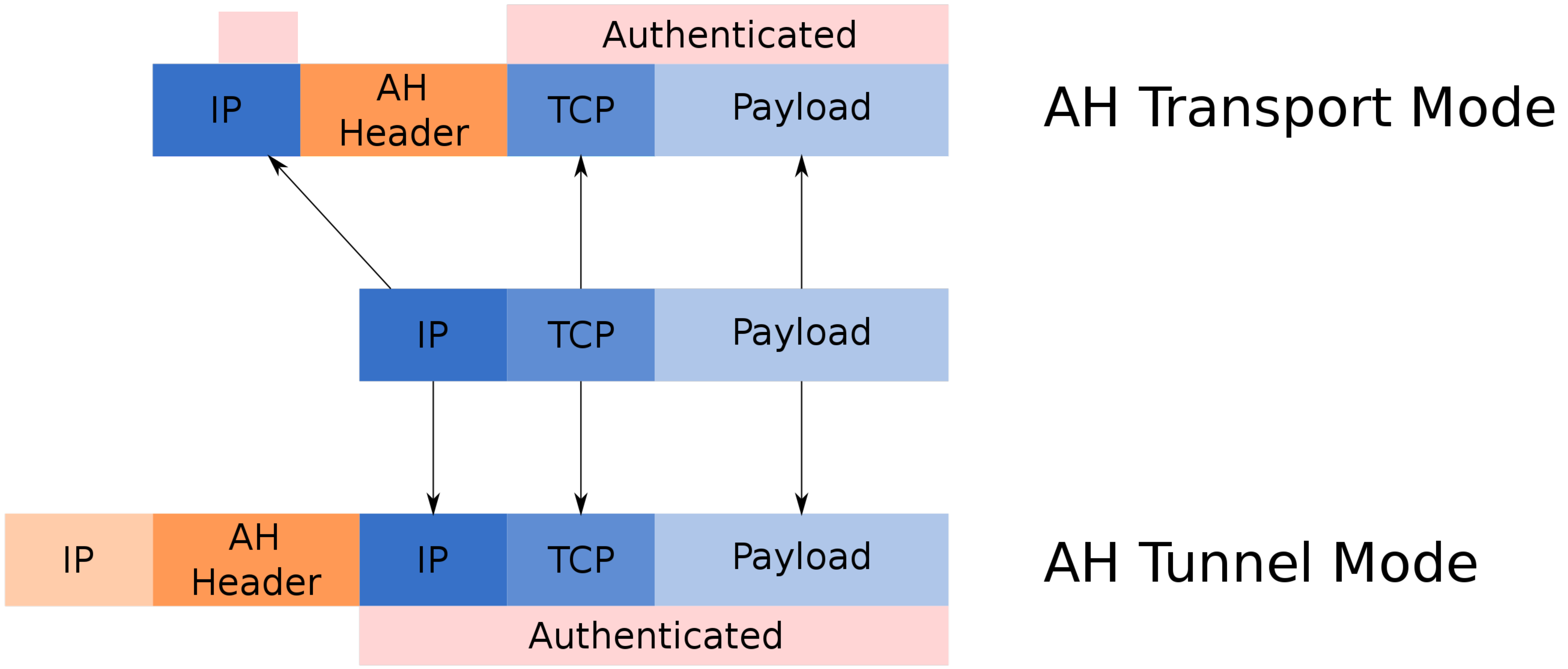


Figure 10.11: IPsec Authentication Header (AH) field view, for both transport and tunnel modes. `Next_header` identifies the protocol of the AH payload (e.g., TCP=6). `Payload_len` is used to calculate the length of the AH header. `SPI` identifies the Security Association. `Sequence_number` allows replay protection (if enabled).

Authentication Header (AH)



Encapsulated Security Payload (ESP)

- Offers confidentiality
  - IP data is encrypted during transmission
- Offers authentication functionality similar to AH
  - But authenticity checks only focus on the IP payload
- Applies padding and generates dummy traffic
  - Makes traffic analysis harder

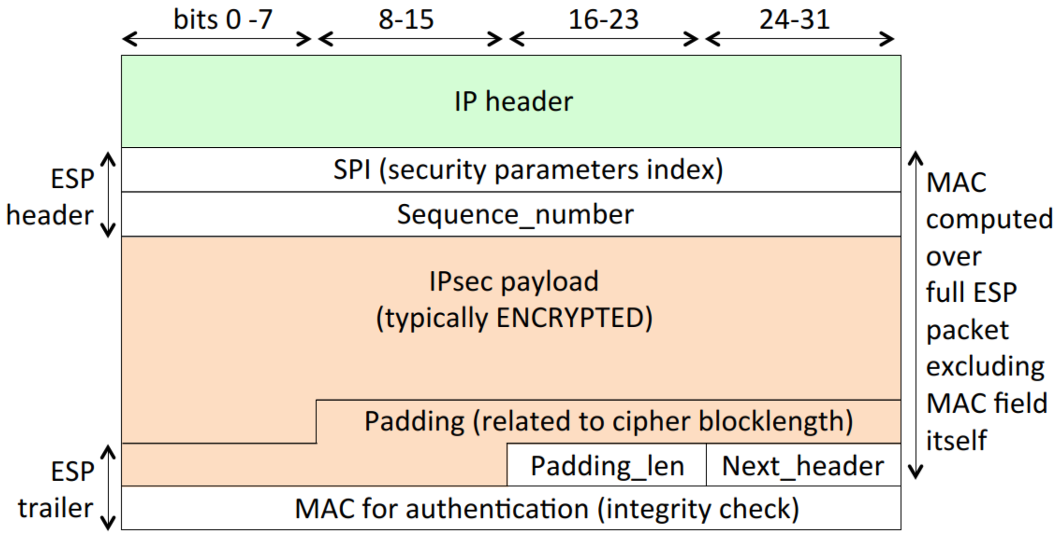
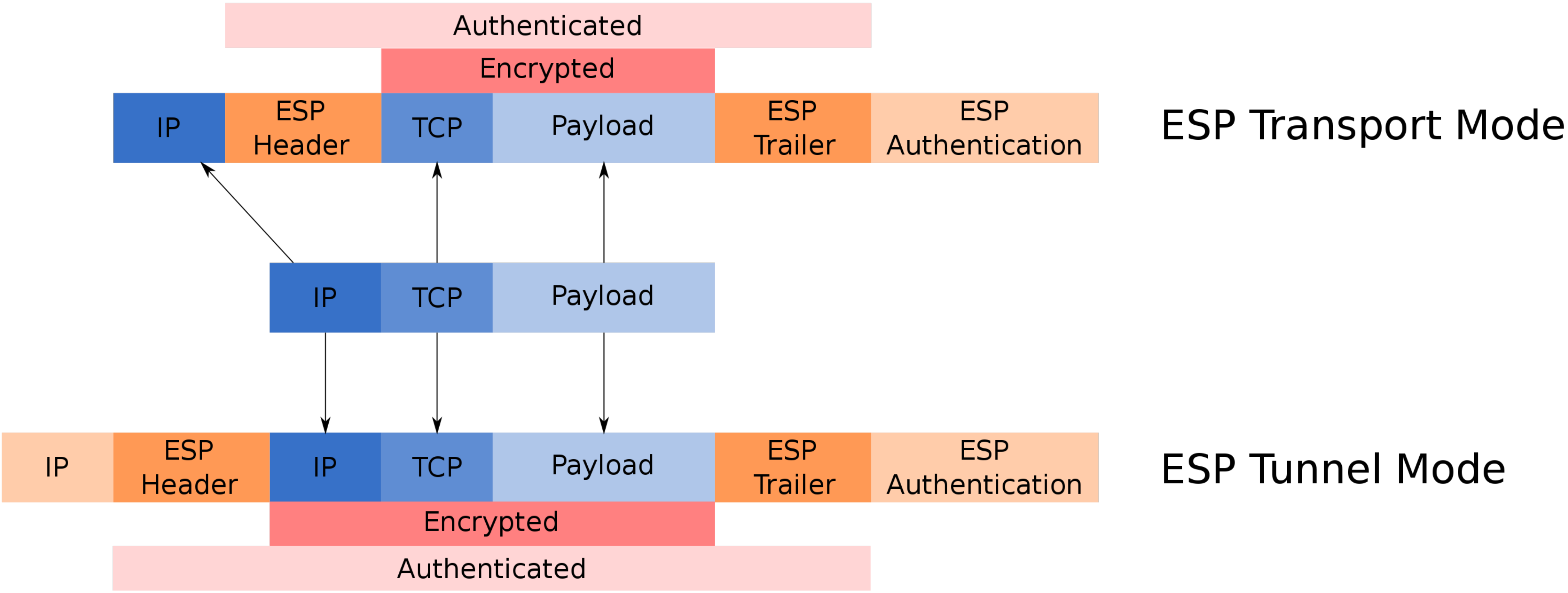


Figure 10.12: IPsec Encapsulating Security Payload (ESP) field view, for both transport and tunnel modes. `SPI` identifies the Security Association. `Sequence_number` allows replay protection (if enabled). `Next_header` (which may include a crypto IV or Initialization Vector) indicates the type of data in the ENCRYPTED field. A payload length field is not needed, as the ESP header is fixed at two 32-bit words, and the length of the IPsec payload (which is the same as that of the original payload) is specified in the IP header.

Encapsulated Security Payload (ESP)



IPSec Deployment Challenges

- Needs to be included in the kernel's network stack.
- There may be legitimate reasons to modify some IP header fields; IPSec breaks networking functionalities that require such changes.
    - E.g., with AH, you cannot replace a private address for a public one at a NAT box.
- IPSec is complex, hard to audit, and prone to misconfigurations

OpenVPN

- Similar to IPSec in tunnel mode but uses OpenSSL for the crypto
- Slower as it is implemented at the user level, not at the kernel level
- Similar security and similar encryption algorithms

Wireguard

- New (and simpler) VPN design built from the ground-up
- Offers a kernel and a user-space implementation
- Faster than IPSec and TLS-based VPN solutions

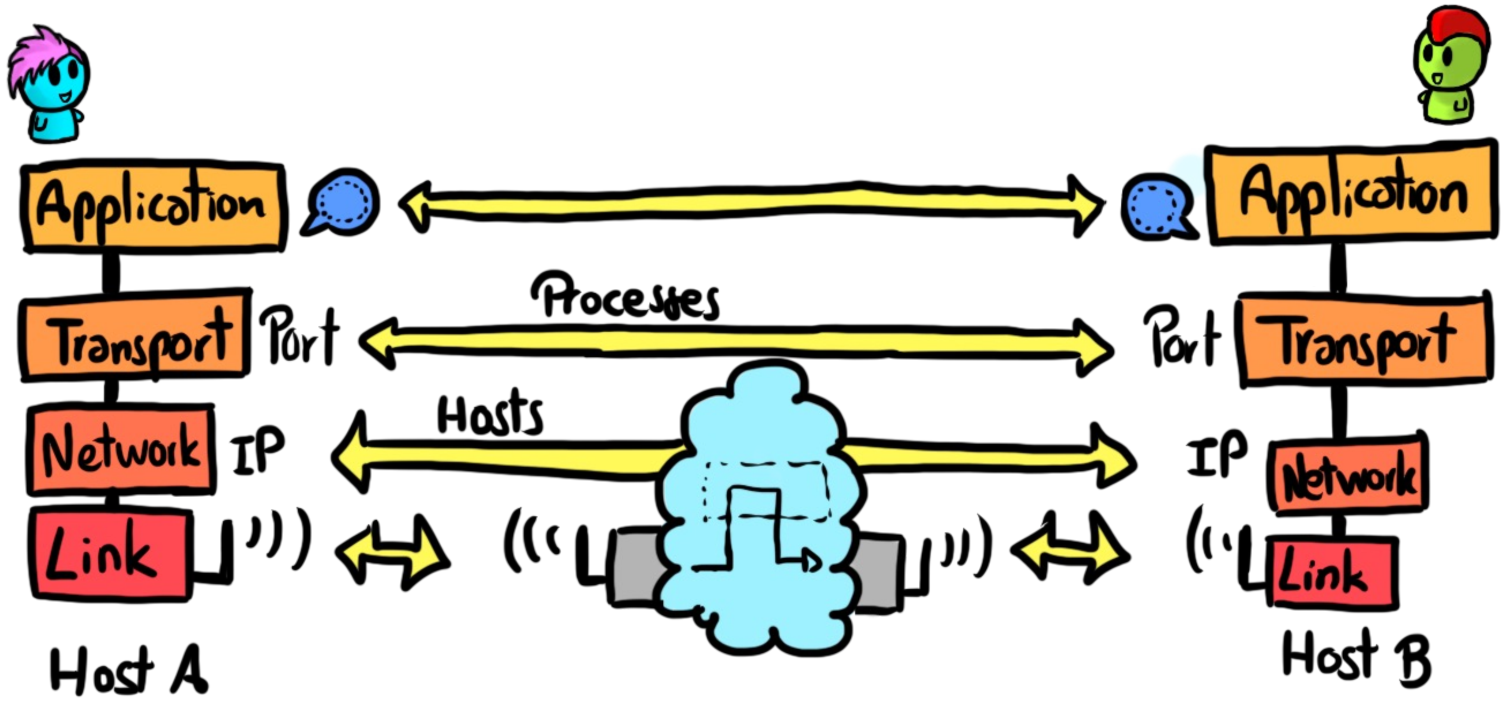
Wireguard

- Easy to configure
  - But no PKI, keys are distributed manually
- Easy to audit
  - 4,000 LoCs vs IPSec's 400,000 LoCs
- Hard to get it wrong
  - Single cipher suite

Transport Layer: TLS

Security through the layers

- Link
  - WEP, WPA, WPA2
- Network
  - VPN, IPsec
- Transport
  - TLS/SSL
- Application
  - ssh, (Next class: PGP, OTR, Signal)



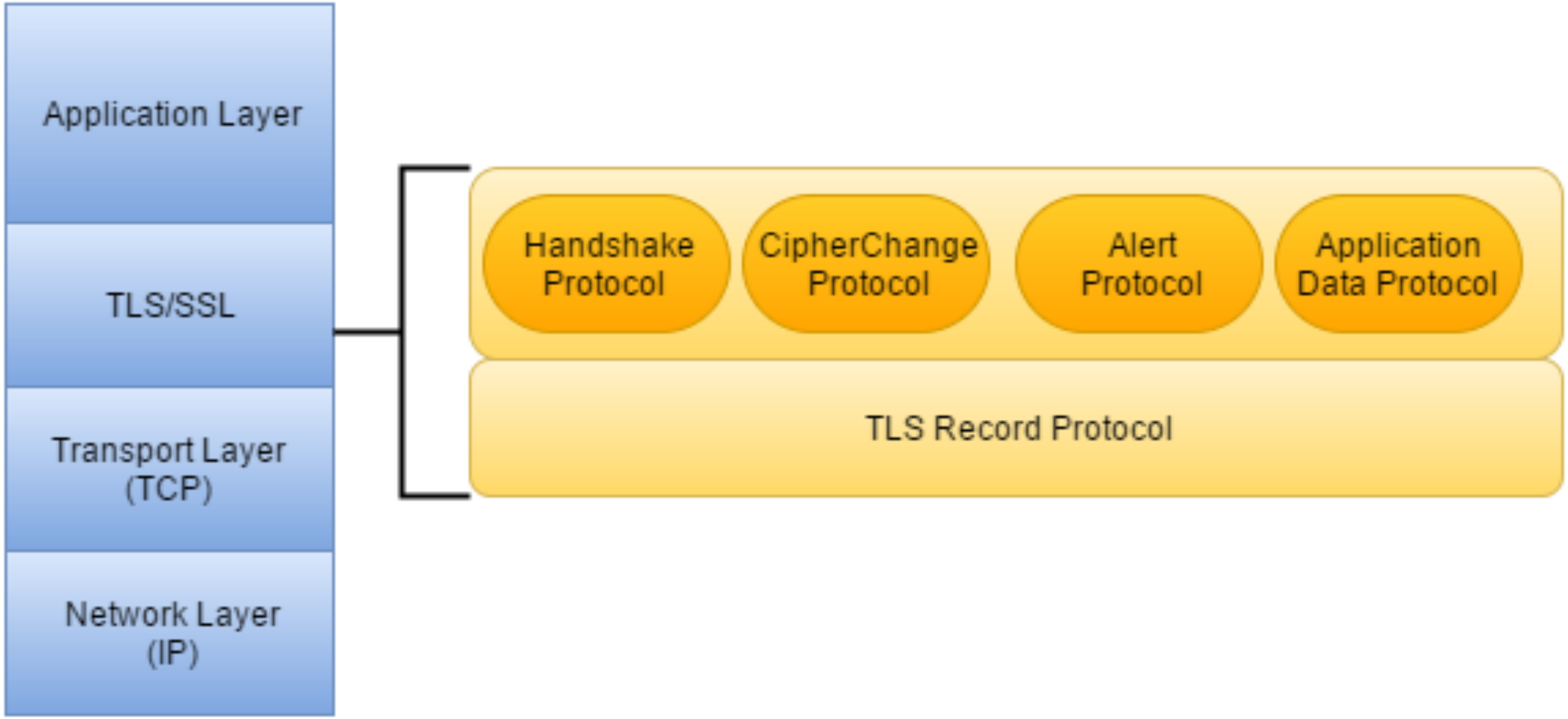
Transport Layer Security Purpose

- Closer to end-to-end security: Client to server
- Defense-in-depth when used in conjunction with IPSec.
  - **Network-layer** security mechanisms arrange to send individual IP packets securely from one network to another
  - **Transport-layer** security mechanisms transform TCP connections to add security and privacy

TLS / SSL

- In the mid-1990s, Netscape invented a protocol called Secure Sockets Layer (SSL) meant for protecting HTTP (web) connections
    - The protocol, however, was general, and could be used to protect any TCP-based connection
    - HTTP + SSL = HTTPS
- Historical note: there was a competing protocol called S-HTTP. But Netscape and Microsoft both chose HTTPS and it endured
- SSL went through a few revisions, and was standardized into the protocol known as TLS (Transport Layer Security)

Where does TLS sit on the network stack?



Source: Vidhatha Vivekananda

TLS Record Protocol

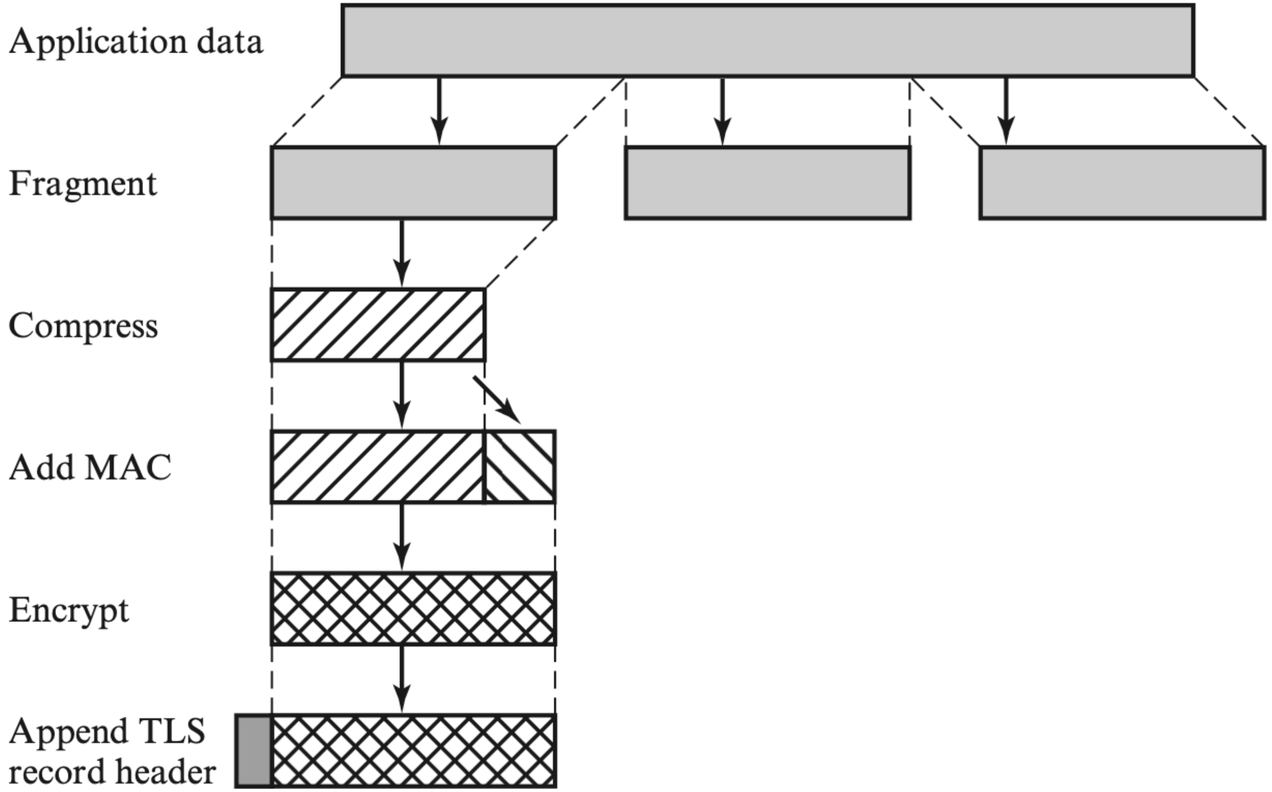


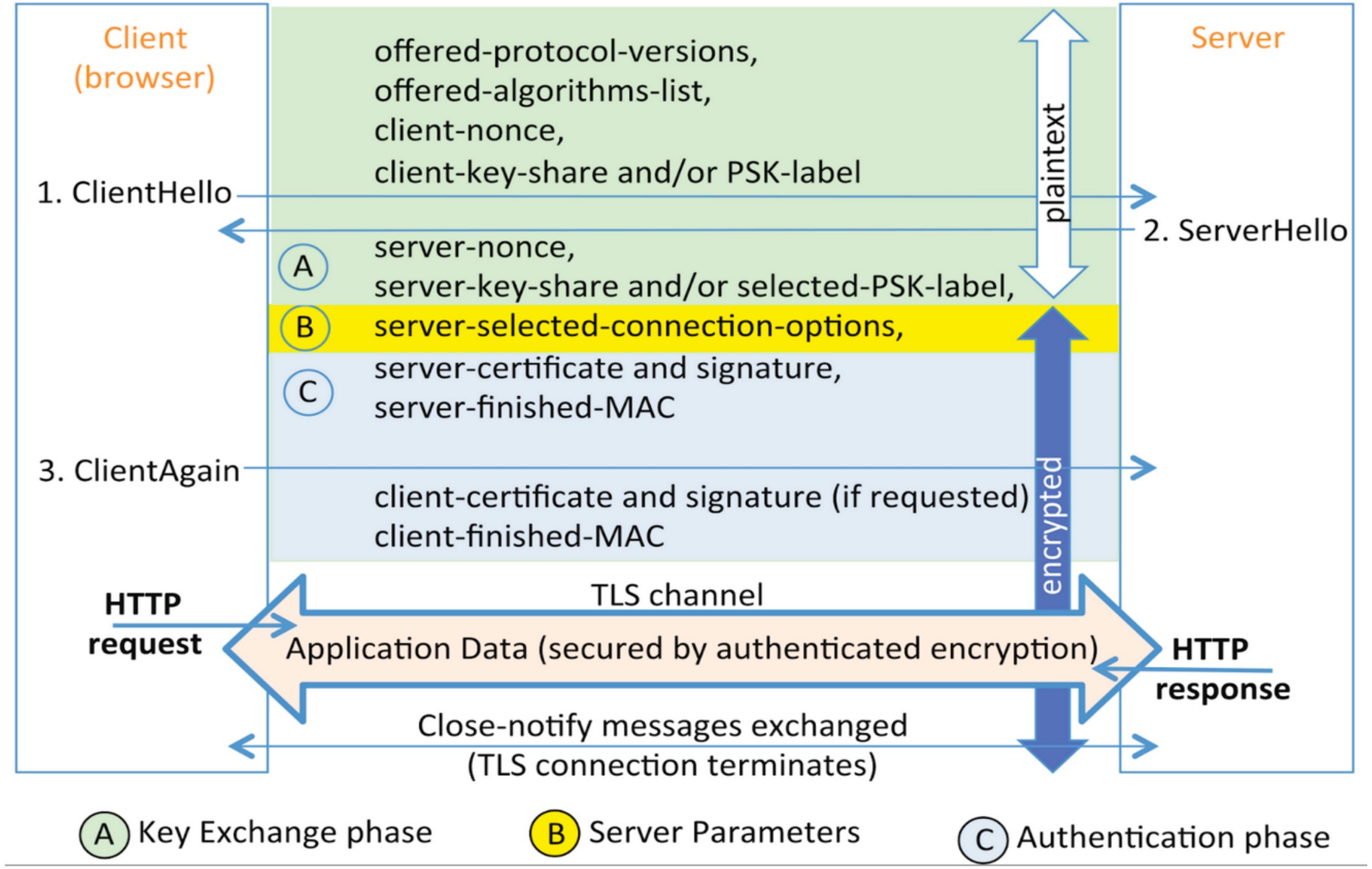
Figure 6.3 TLS Record Protocol Operation

The TLS Handshake – To Establish Sessions

The client and server will do the following to **establish a session**:

- Specify which version of TLS they will use
- Decide on which cipher suites they will use
    - Typically, AES and SHA256
- Authenticate the identity of the server via the server's public key and the SSL certificate authority's digital signature
- Generate session keys in order to use symmetric encryption after the handshake is complete

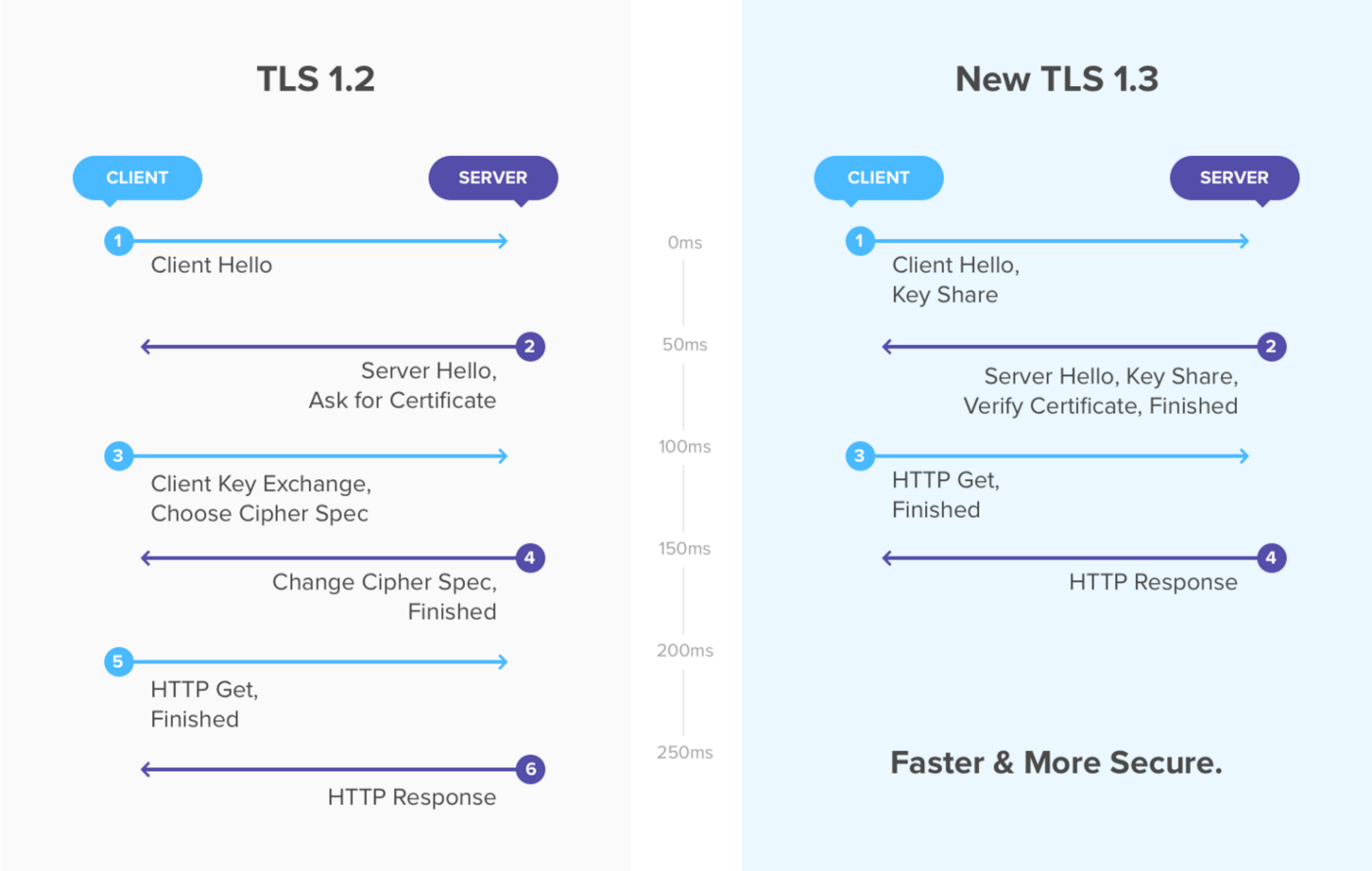
TLS 1.3 Handshake



TLS 1.2 vs TLS 1.3

TLS 1.3:

- Less trips
- Deprecate older ciphers
- Removed compression



Source: Bidhan Khatri

TLS Design Choices

- Highly configurable protocol with many options/versions
  - Different authentication/key exchange protocols
  - Different encryption and signature algorithms
- Authentication
  - Usually (!) one-sided, only server authenticates
  - Server PKI certificates
  - Secure Channel: Software distribution
- Hybrid encryption (symmetric for data, asymmetric for key exchange)
  - Key Exchange: Authenticated Diffie-Hellman

CAs in TLS

A certification authority acts as a trusted third-party that:

- Issues digital certificates
- Certifies the ownership of a public key by the subject of the certificate
- Manages certificate revocation lists (CRLs)

Why one-sided authentication?

- PKI is a “somewhat” closed system
  - Difficult to obtain certificate
  - Difficult to manage keys
  - User-unfriendly
- PKI secure channel is “somewhat” easy to implement
  - Certificate authorities pay software vendors
  - Certificate authorities can be malicious/broken
- Web traffic contains “few” servers and many clients
  - Efficient way of implementing authentication

Preventing Modifications by Mallory

- Authenticated Encryption
  - MACs with every “packet”
  - Mallory cannot modify packet
- Can Mallory drop a packet?
  - MAC is dropped alongside with it
  - Nothing to verify
- Can Mallory replay a packet?
  - MAC is correct
  - Solution: **Sequence Numbers**

Doesn't TLS suffice?

Or, why do I need Link or Network layer protection anymore?

- TLS only encrypts the payload, not source/destination IP.
- Still don't want to expose internal network via Wi-Fi.
- Redundancy!

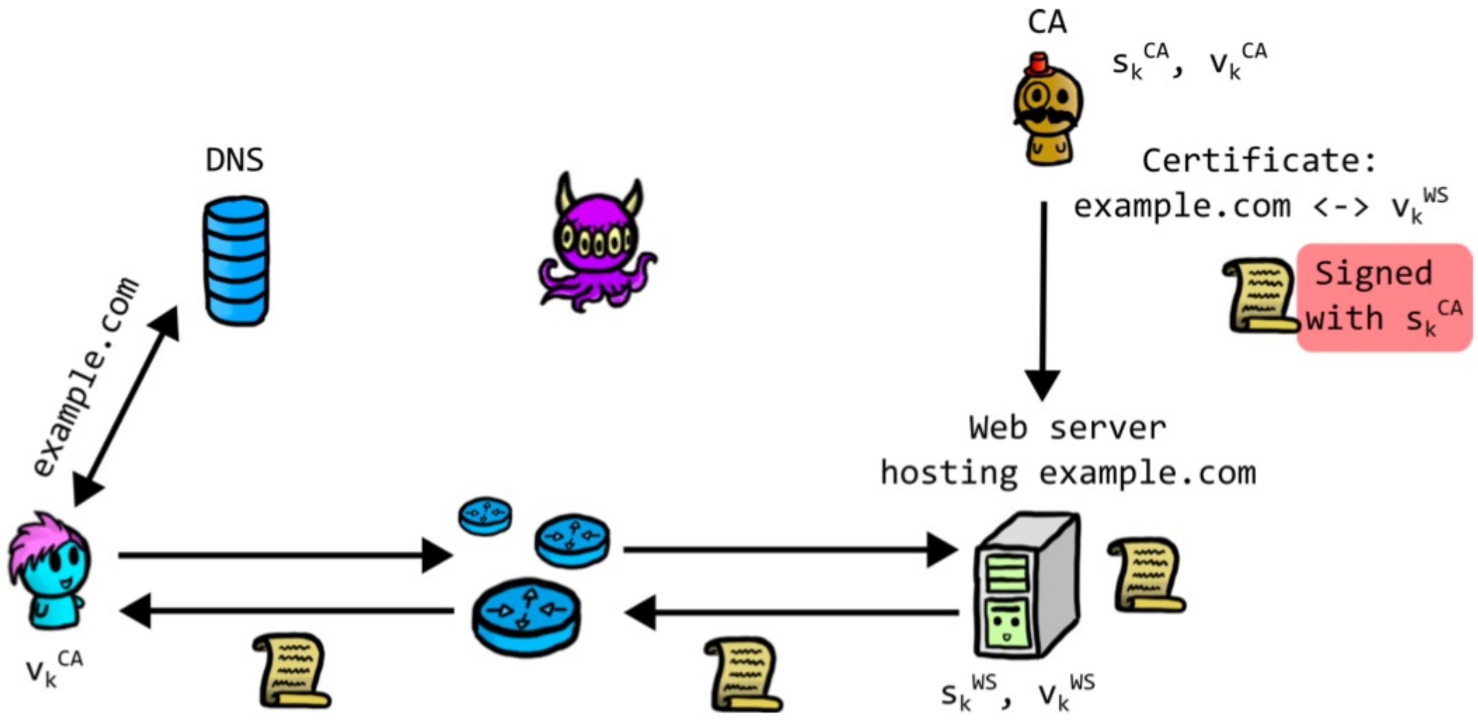
What can go wrong with TLS

- Implementation issues
- Compromising CAs
- Using weak ciphers

Recall TLS Authentication

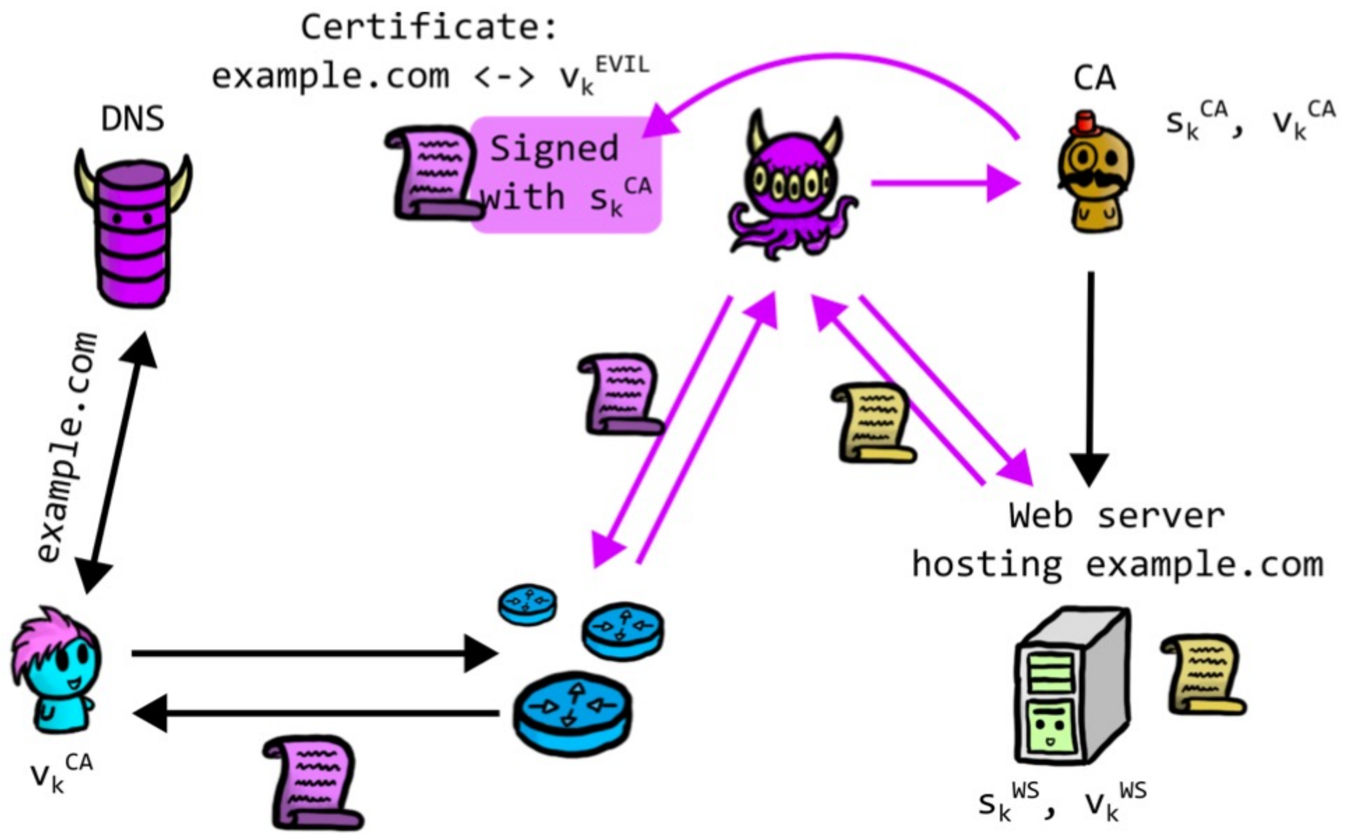
Basic idea: Alice accepts the connection if she receives a certificate and

1. the certificate is signed by a CA she trusts (v_k^{CA})
2. the certificate is for the domain she's requesting
3. when talking to the web server, Alice can verify the signatures with v_k^{WS} (which is in the certificate).



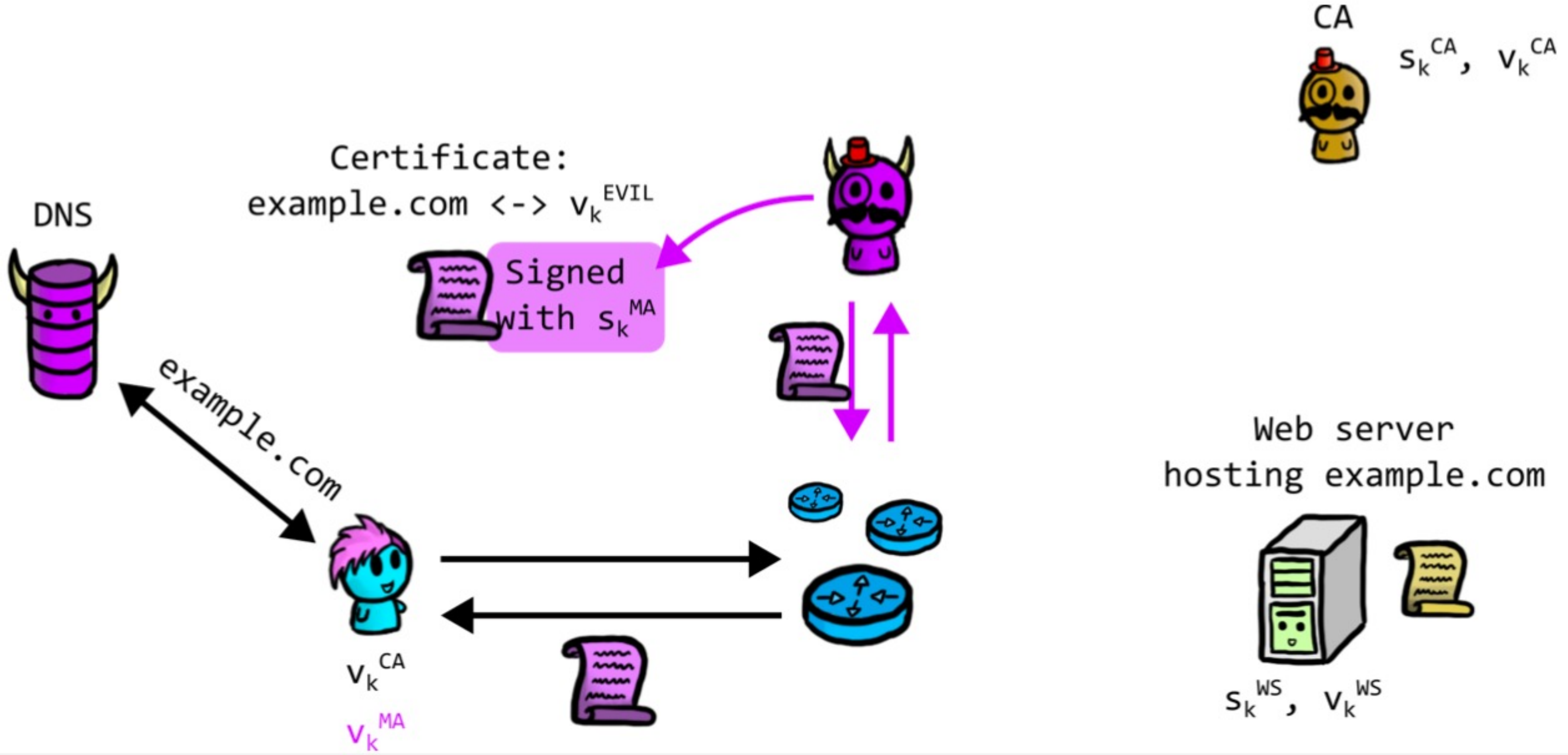
Compromising CAs – Single Shot

An adversary can compromise a CA to plant fake certificates (e.g., DigiNotar's fake *.google.com certificates used by an ISP in Iran)



Compromising CAs – Unlimited

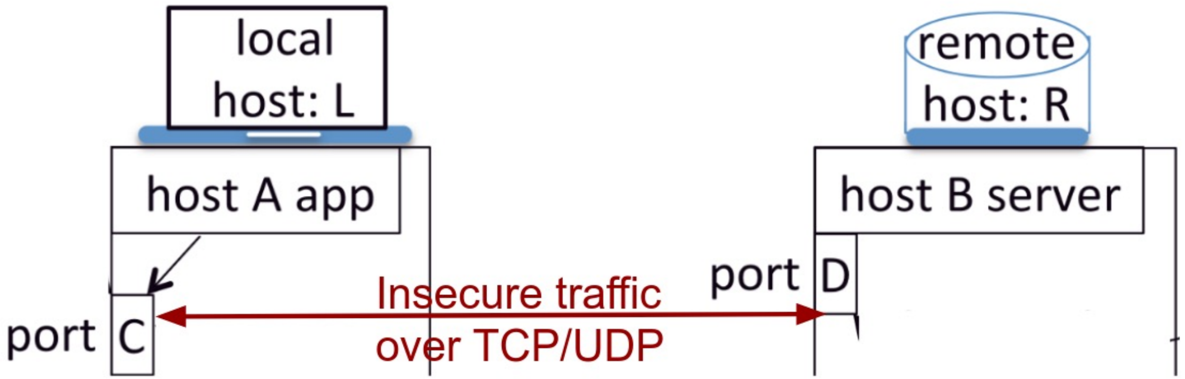
An adversary can install a custom CA on users' devices, allowing them to sign certificates that clients will accept for any site (e.g., in 2019, Kazakhstan's ISPs mandated the installation of a root certificate issued by the government)



Application Layer - SSH

Pre-SSH

- Suppose that you want to connect to a remote machine
  - You may think “Oh ok, let me use Telnet”
- Think again...
  - All data exchanged through Telnet is in plain text!



Enter Secure Remote Login - SSH

Usage (simplified):

- Client connects to server
- Server sends its verification key
    - The client should verify that this is the correct key
    - Many clients implement trust on first use (TOFU)
- Client and server run a key agreement protocol to establish session keys, server signs its messages
    - All communication from here on in is encrypted and MAC-ed with the session keys
- Client authenticates to server
- Server accepts authentication, login proceeds

How does the client authenticate

There are two main ways to authenticate with ssh:

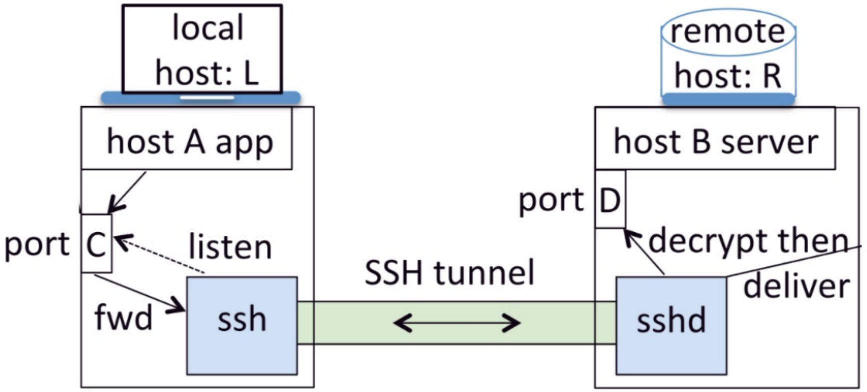
- Send a password over the encrypted channel
  - The server needs to know (a hash of) your password
- Sign a random challenge with your private signature key
  - The server needs to know your public verification key

Q: Advantages / Disadvantages of each

SSH Port Forwarding

SSH allows for tunneling:

- The client machine can create a mapping between a local TCP port and a port in the remote machine
  - e.g., localhost:IMAP to mail.myorg.ca:IMAP
- The client SSH and the server SSHd operate as a secure relay
  - Allows the client to interact with server applications via SSH



Next Class: PGP, OTR, Signal