Security: Securing Protocols, HTTPS, IPSec, Operational Security, Firewalls

David Marchant

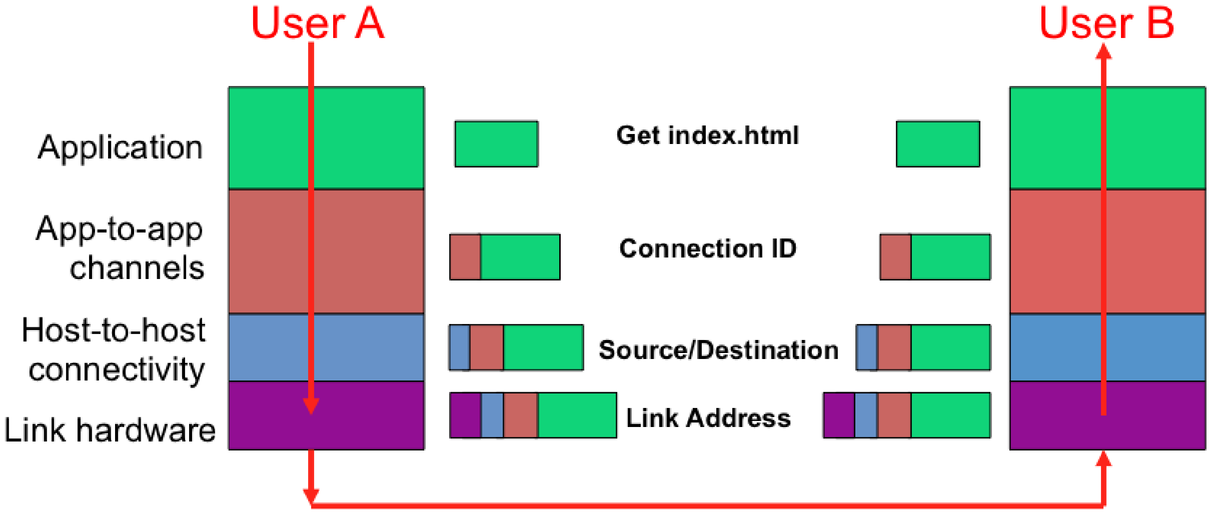
Based on slides compiled by Marcos Vaz Salles with modifications by Vivek Shah

What should you be able to do after today?

- List security properties and related attacks
- Relate basic cryptographic schemes to their use in network protocols
- Explain the main mechanisms of HTTPS
- Explain the motivation and uses of IPSec
- Discuss operational security concerns and solutions, such as firewalls

Do-It-Yourself Recap: HTTP

- What transport protocol does HTTP use? Can an unauthorized party read the content of HTTP requests/responses?
- Which two types of performance optimizations are common with HTTP and web applications?



Source: Freedman (partial)

Nothing is secure forever

“You have 1 minute to design a maze that takes 2 minutes to solve” – some scriptwriter

Source: Freedman (partial)

Internet’s Design: Insecure

- Designed for simplicity
- “On by default” design
- Readily available zombie machines
- Attacks look like normal traffic
- Internet’s federated operation obstructs cooperation for diagnosis/mitigation

Source: Freedman (partial)

Security Properties

- **Confidentiality:** Concealment of information or resources
- **Authenticity:** Identification and assurance of origin of info
- **Integrity:** Trustworthiness of data or resources in terms of preventing improper and unauthorized changes
- **Availability:** Ability to use desired info or resource
- **Non-repudiation:** Offer of evidence that a party indeed is sender or a receiver of certain information

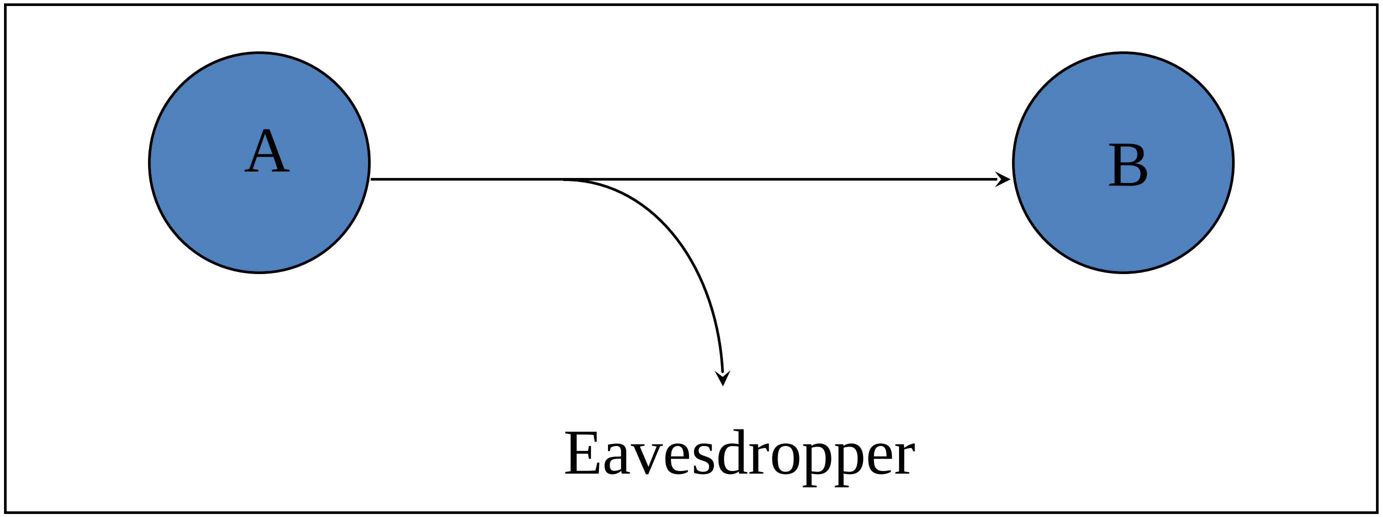
---

- **Access control:** Facilities to determine and enforce who is allowed access to what resources (host, software, network, …)

Source: Freedman (partial)

Eavesdropping - Message Interception (Attack on Confidentiality)

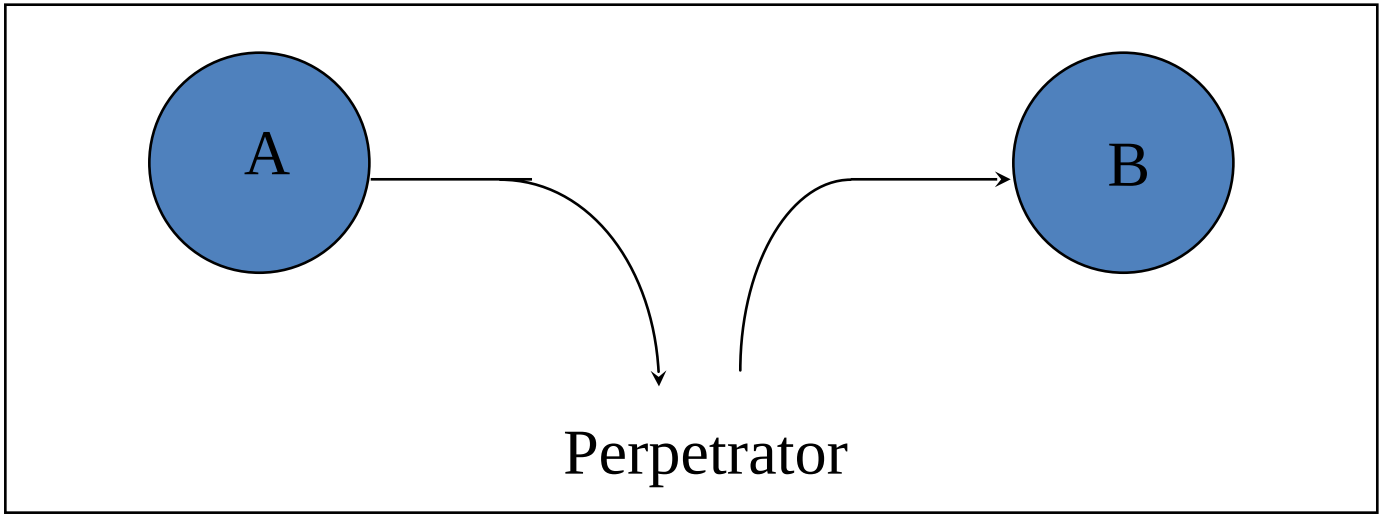
- Unauthorized access to information
- Packet sniffers and wiretappers (e.g. tcpdump)
- Illicit copying of files and programs



Source: Freedman

Integrity Attack - Tampering

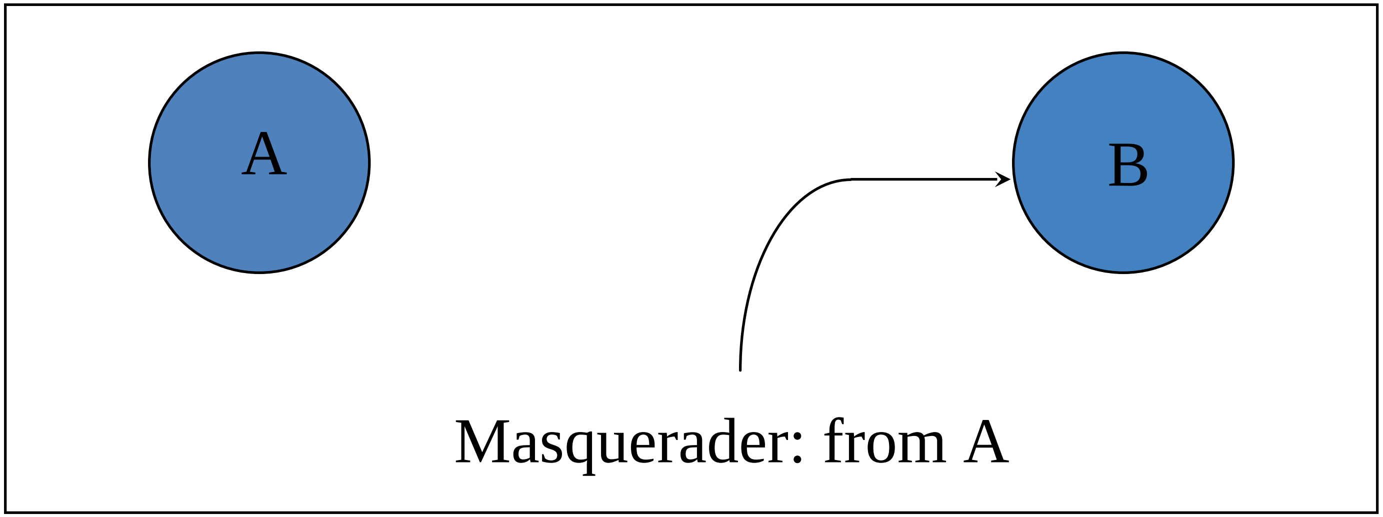
- Stop the flow of the message
- Delay and optionally modify the message
- Release the message again



Source: Freedman

Authenticity Attack - Fabrication

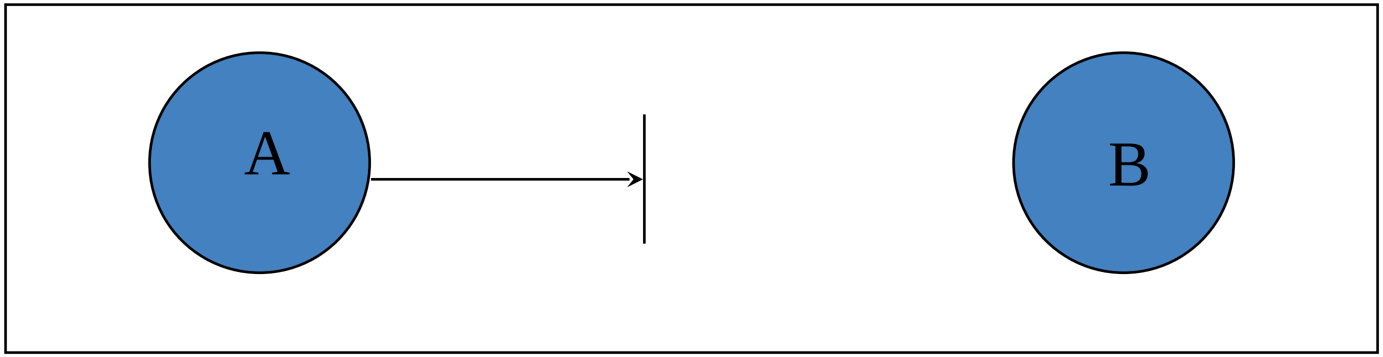
- Unauthorized assumption of other’s identity
- Generate and distribute objects under identity



Source: Freedman

Attack on Availability

- Destroy hardware (cutting fiber) or software
- Modify software in a subtle way
- Corrupt packets in transit



- Blatant *denial of service* (DoS):
  - Crashing the server
  - Overwhelm the server (use up its resource)

Source: Freedman

Impact of Attacks

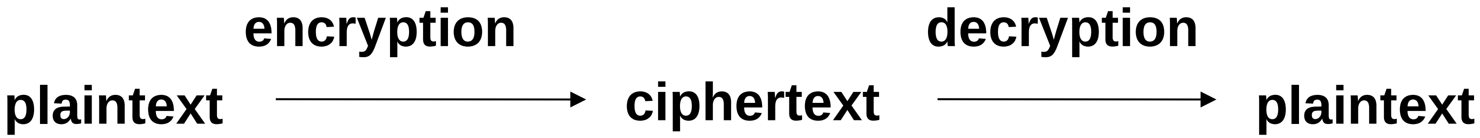
- Theft of confidential information
- Unauthorized use of
  - Network bandwidth
  - Computing resource
- Spread of false information
- Disruption of legitimate services

Source: Freedman

What is Cryptography?

- Comes from Greek word meaning “secret”
  - Primitives also can provide integrity, authentication

- Cryptographers invent secret codes to attempt to hide messages from unauthorized observers



- Modern encryption:
  - *Algorithm* public, *key* secret and provides security
  - May be symmetric (secret) or asymmetric (public)

Source: Freedman

Cryptography != Hashing

- Hashing is the practice of transforming data to a unique (ish) value of fixed length
- In hashing it is impossible to reverse engineer (Though usually we don’t need to)

Password → g8Dl9w

- Cryptography is the practice of altering the content of a message in some way to obscure its meaning
- Cryptogaphy it is impossible to make something that cannot be reverse engineered (Though thats actually what we want)

Password → pASSWORD

Source: Freedman

Three Types of Functions

- Cryptographic hash Functions
  - Zero keys
  - Not sufficently secure
  - Very quick
- Secret-key functions (Symetric key)
  - One key
  - Very secure, very difficult to distribute
  - Quick, compared to Public Keys
- Public-key functions (Asymetric key)
  - Two keys
  - Very secure, easy to distribute
  - Very slow

Source: Freedman

Use of encryption and MAC/signatures

Confidentiality (Encryption)

Sender:

- Compute C = $Enc_K(M)$
- Send C

Receiver:

- Recover M = $Dec_K(C)$

Auth/Integrity (MAC / Signature)

Sender:

- Compute s = $Sig_K(Hash (M))$
- Send <M, s>

Receiver:

- Computer s' = $Ver_K(Hash (M))$
- Check s' == s

These are simplified forms of the actual algorithms

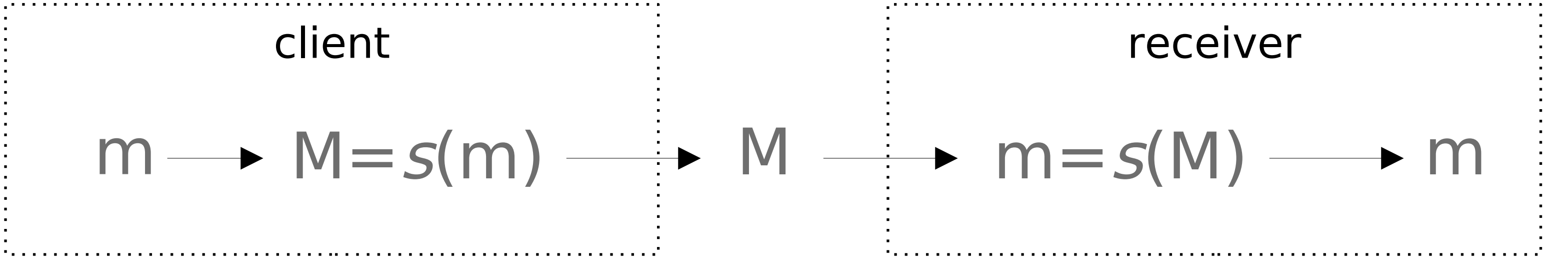
Secrets are often added to make these more secure

Source: Freedman

Using Keys

- Secret-key functions

  secret *s* must be known by both ahead of time



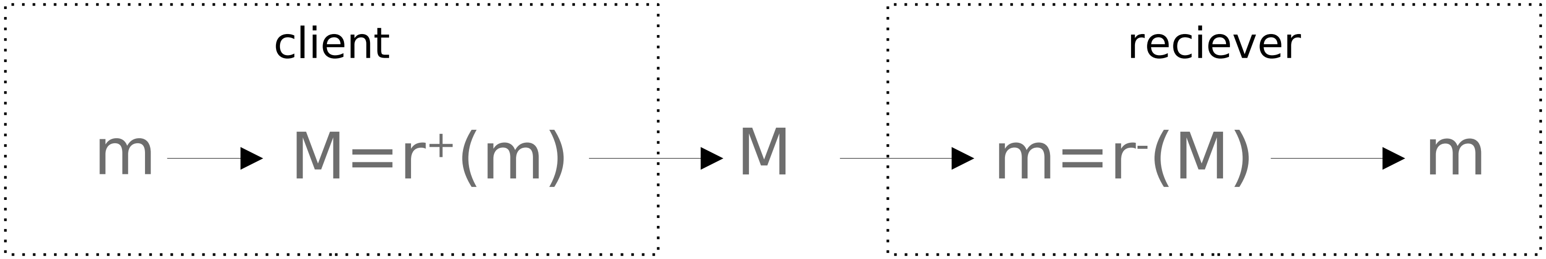
client

receiver

$m \rightarrow M=s(m) \rightarrow M \rightarrow m=s(M) \rightarrow m$

- Public-key functions

  reciever has a public key r_+ and private key r_-



client

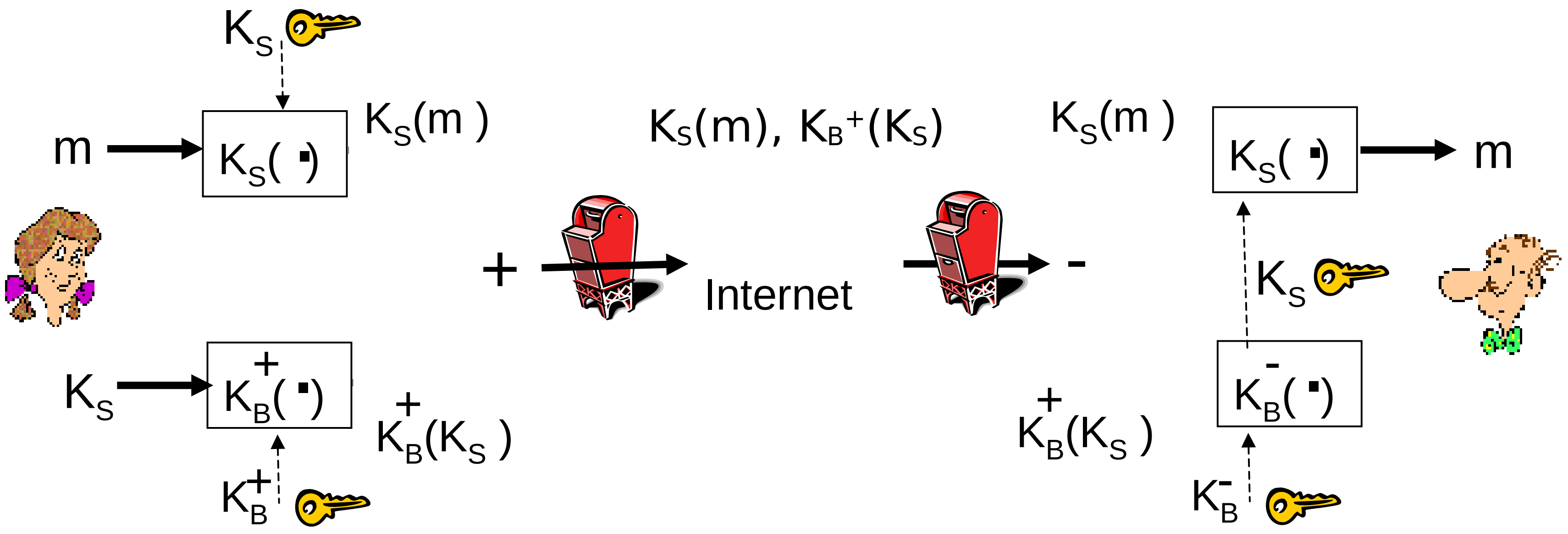
reciever

$m \rightarrow M=r^+(m) \rightarrow M \rightarrow m=r^-(M) \rightarrow m$

Source: Freedman

Secure Email

Alice wants to send confidential e-mail, m, to Bob.

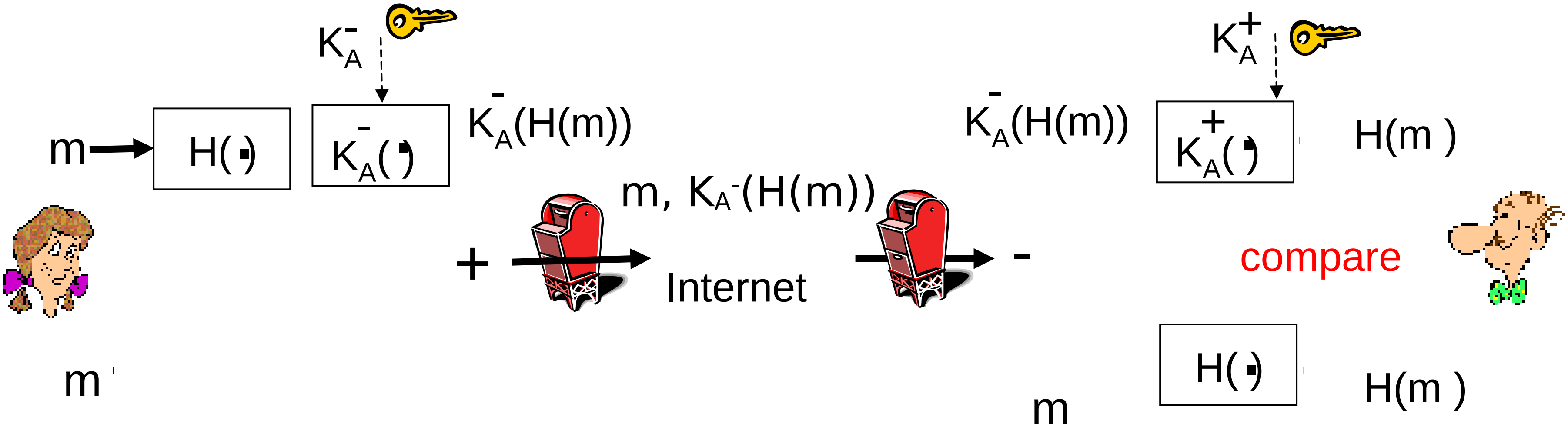


Use a random secret to encrypt and send that too.

Source: Kurose & Ross

Secure Email (continued)

Alice wants to provide sender authentication message integrity

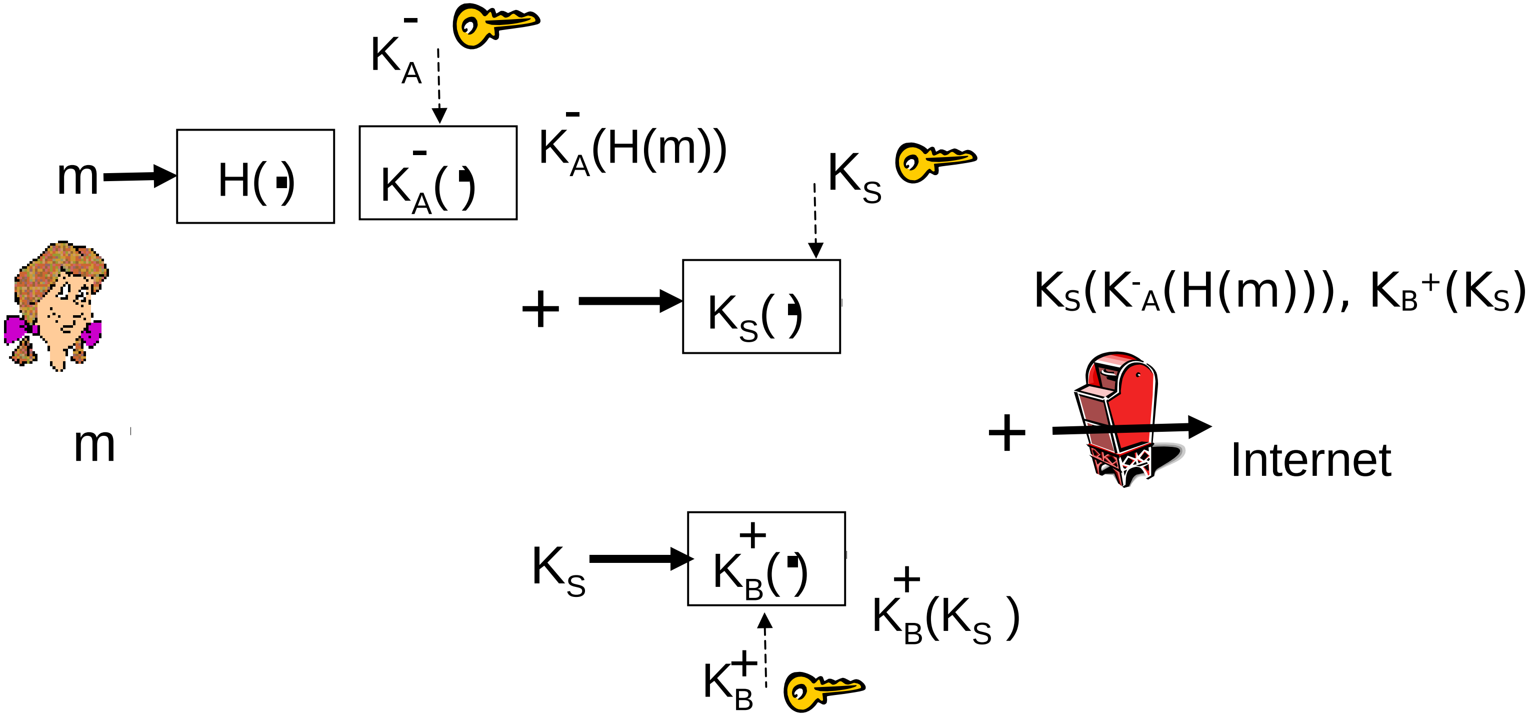


Use a private key to encrypt the hash of the message.

Source: Kurose & Ross

Secure Email (continued)

Alice wants to provide secrecy, sender authentication, message integrity.



Use both approaches in combination.

Source: Kurose & Ross

Public / Private Keys

- Not just used for email as a hidden feature
- Used commonly across many networked systems, some of which you may have used
- Github, SSH, ERDA
- Each system only as secure as you are...

Source: Kurose & Ross

HTTP and security

- Assume you have the cryptographic primitives:
  - Confidentiality: Symmetric cryptography (key K)
    - $C = Enc_{K}(M)$
    - $M = Dec_{K}(C)$
  - Confidentiality: Asymmetric (“public-key”) cryptography (keys PK/SK)
    - $C = Enc_{PK}(M)$
    - $M = Dec_{SK}(C)$
  - Authentication: Asymmetric (“public-key”) cryptography (keys PK/SK)
    - $s = Sig_{SK}(Hash\ (M))$
    - $s' = Ver_{PK}(Hash\ (M))$
    - Check s’ == s
- How do you secure HTTP?
  - Ensure sites are the sites you really requested
  - Ensure no one else can read or forge requests/responses

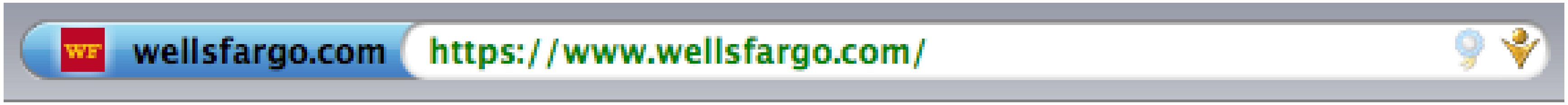
"Securing" HTTP

- Threat model
  - Eavesdropper listening on conversation (confidentiality)
  - Man-in-the-middle modifying content (integrity)
  - Adversary impersonating desired website (authentication, and confidentiality)

- Enter HTTP-S
  - HTTP sits on top of secure channel (SSL/TLS)
  - All (HTTP) bytes written to secure channel are encrypted and authenticated
  - Problem: What is actually authenticated to prevent impersonation? Which keys used for crypto protocols?

Source: Freedman

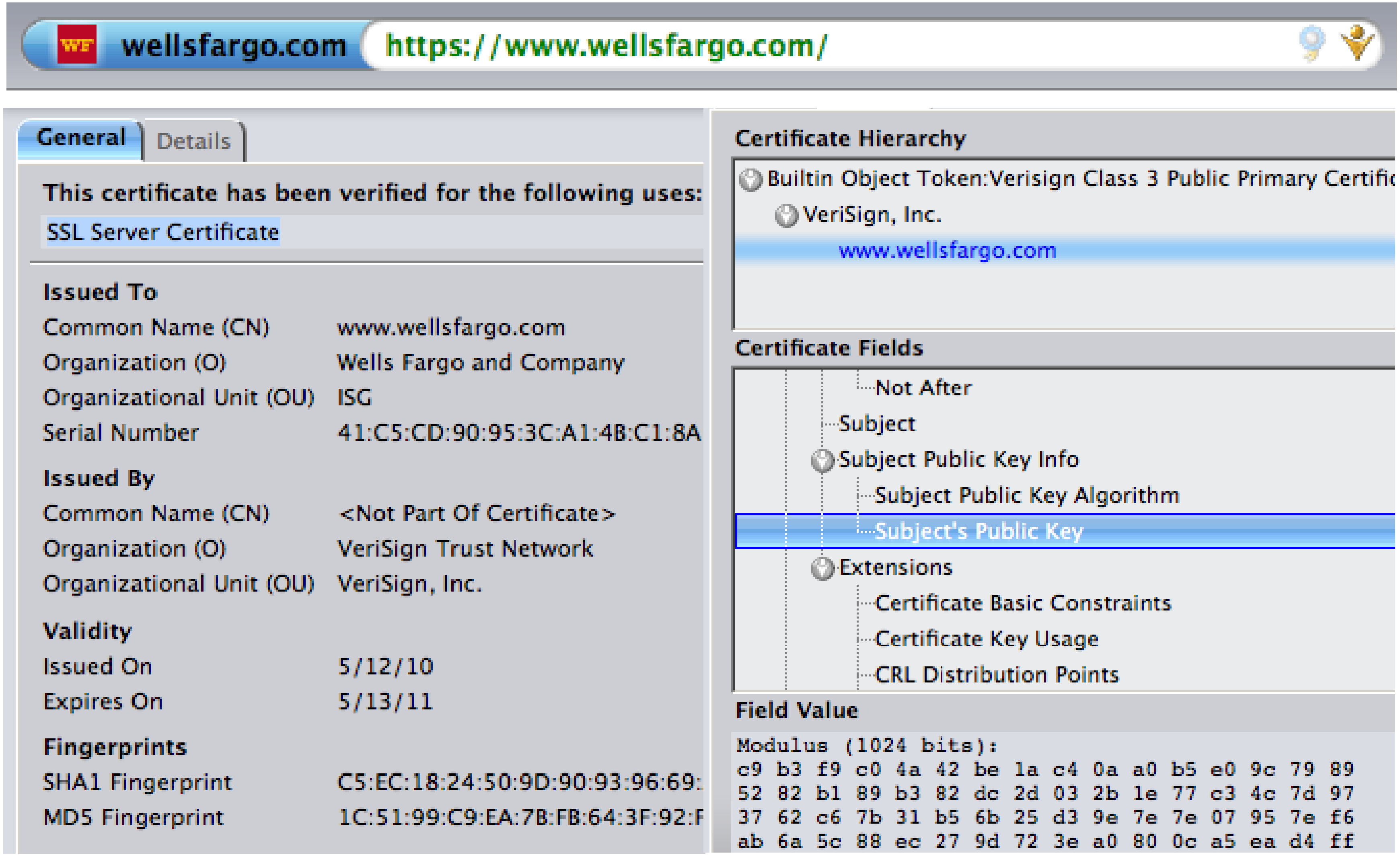
Learning a valid public key



- What is that lock?
  - Securely binds domain name to public key (PK)
    - Believable only if you trust the attesting body
    - Bootstrapping problem: Who to trust, and how to tell if this message is actually from them?
  - If PK is authenticated, then any message signed by that PK cannot be forged by non-authorized party

Source: Freedman

How to authenticate PK



https://www.wellsfargo.com/

This certificate has been verified for the following uses:

SSL Server Certificate

Issued To

Common Name (CN)	www.wellsfargo.com
Organization (O)	Wells Fargo and Company
Organizational Unit (OU)	ISG
Serial Number	41:C5:CD:90:95:3C:A1:4B:C1:8A

Issued By

Common Name (CN)	<Not Part Of Certificate>
Organization (O)	VeriSign Trust Network
Organizational Unit (OU)	VeriSign, Inc.

Validity

Issued On	5/12/10
Expires On	5/13/11

Fingerprints

SHA1 Fingerprint	C5:EC:18:24:50:9D:90:93:96:69:
MD5 Fingerprint	1C:51:99:C9:EA:7B:FB:64:3F:92:F

Certificate Hierarchy

- Builtin Object Token:Verisign Class 3 Public Primary Certific
  - VeriSign, Inc.
    - www.wellsfargo.com

Certificate Fields

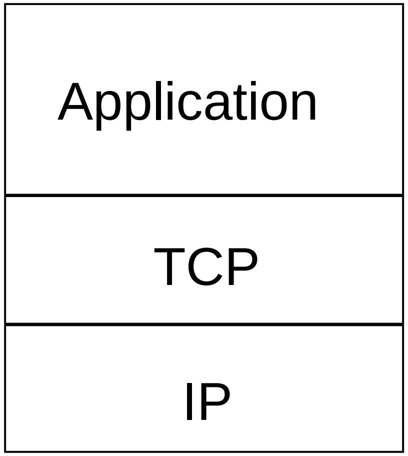
- Not After
- Subject
- Subject Public Key Info
  - Subject Public Key Algorithm
  - Subject's Public Key
- Extensions
  - Certificate Basic Constraints
  - Certificate Key Usage
  - CRL Distribution Points

Field Value

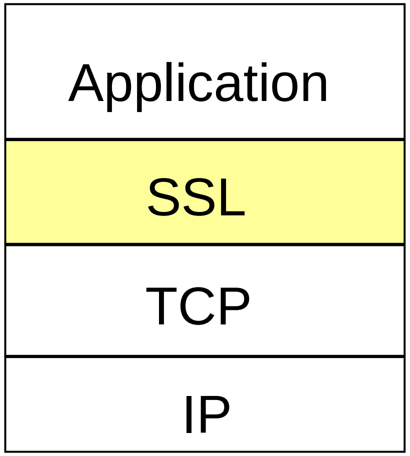
```
Modulus (1024 bits):
c9 b3 f9 c0 4a 42 be 1a c4 0a a0 b5 e0 9c 79 89
52 82 b1 89 b3 82 dc 2d 03 2b 1e 77 c3 4c 7d 97
37 62 c6 7b 31 b5 6b 25 d3 9e 7e 7e 07 95 7e f6
ab 6a 5c 88 ec 27 9d 72 3e a0 80 0c a5 ea d4 ff
```

Source: Freedman

SSL and TCP/IP



normal application

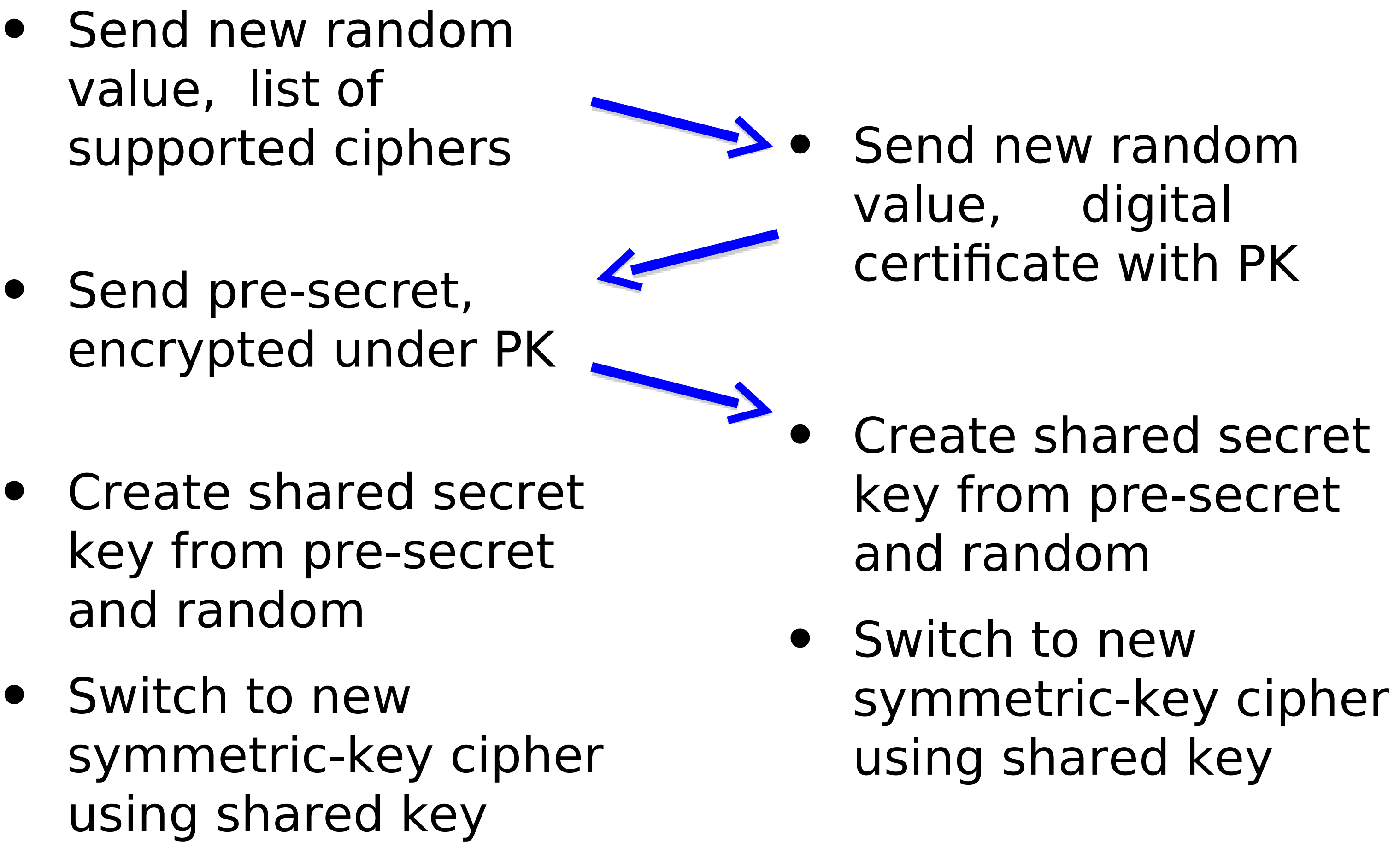


application with SSL

- SSL provides application programming interface (API) to applications.
- C, Python and Java SSL libraries/classes readily available

Source: Kurose & Ross

Transport Layer Security (TLS) – Replaces SSL



- Send new random value, list of supported ciphers
- Send pre-secret, encrypted under PK
- Create shared secret key from pre-secret and random
- Switch to new symmetric-key cipher using shared key

- Send new random value, digital certificate with PK
- Create shared secret key from pre-secret and random
- Switch to new symmetric-key cipher using shared key

Source: Freedman (partial)

Source: Freedman (partial)

Comments on HTTPS

- Note that HTTPS authenticates server, not content
  - If CDN (Akamai) serves content over HTTPS for its customers, customer must trust Akamai not to change content
- Switch to symmetric-key crypto after public-key ops
  - Symmetric-key crypto much faster (100-1000x)
  - PK crypto can encrypt message only approx. as large as key (1024 bits – this is a simplification) – afterwards uses hybrid
- HTTPS on top of TCP, so reliable byte stream
  - Can leverage fact that transmission is reliable to ensure: each data segment received exactly once
  - Adversary can’t successfully drop or replay packets

IP Security

- There are range of app-specific security mechanisms
  - eg. TLS/HTTPS, S/MIME, PGP, Kerberos,
- But security concerns that cut across protocol layers
- Implement by the network for all applications?

Enter IPSec!

Source: Freedman

IPSec

- General IP Security mechanism framework
- Allows one to provide
  - Access control, integrity, authentication, originality, and confidentiality
- Applicable to different settings
  - Narrow streams: Specific TCP connections
  - Wide streams: All packets between two gateways

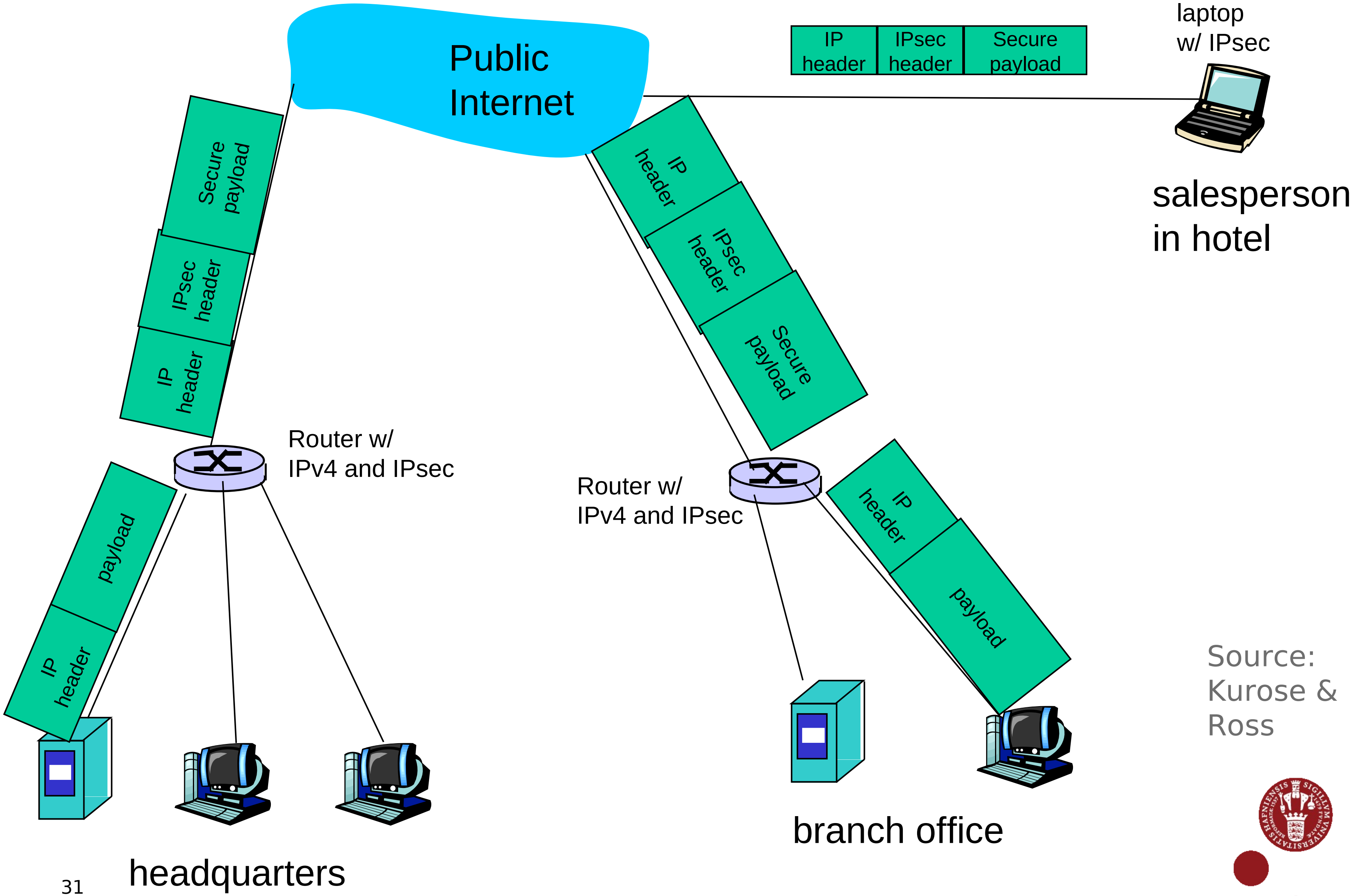
Source: Freedman

Virtual Private Networks (VPNs)

- institutions often want private networks for security.
  - costly: separate routers, links, DNS infrastructure.
- VPN: institution’s inter-office traffic is sent over public Internet instead
  - encrypted before entering public Internet
  - logically separate from other traffic

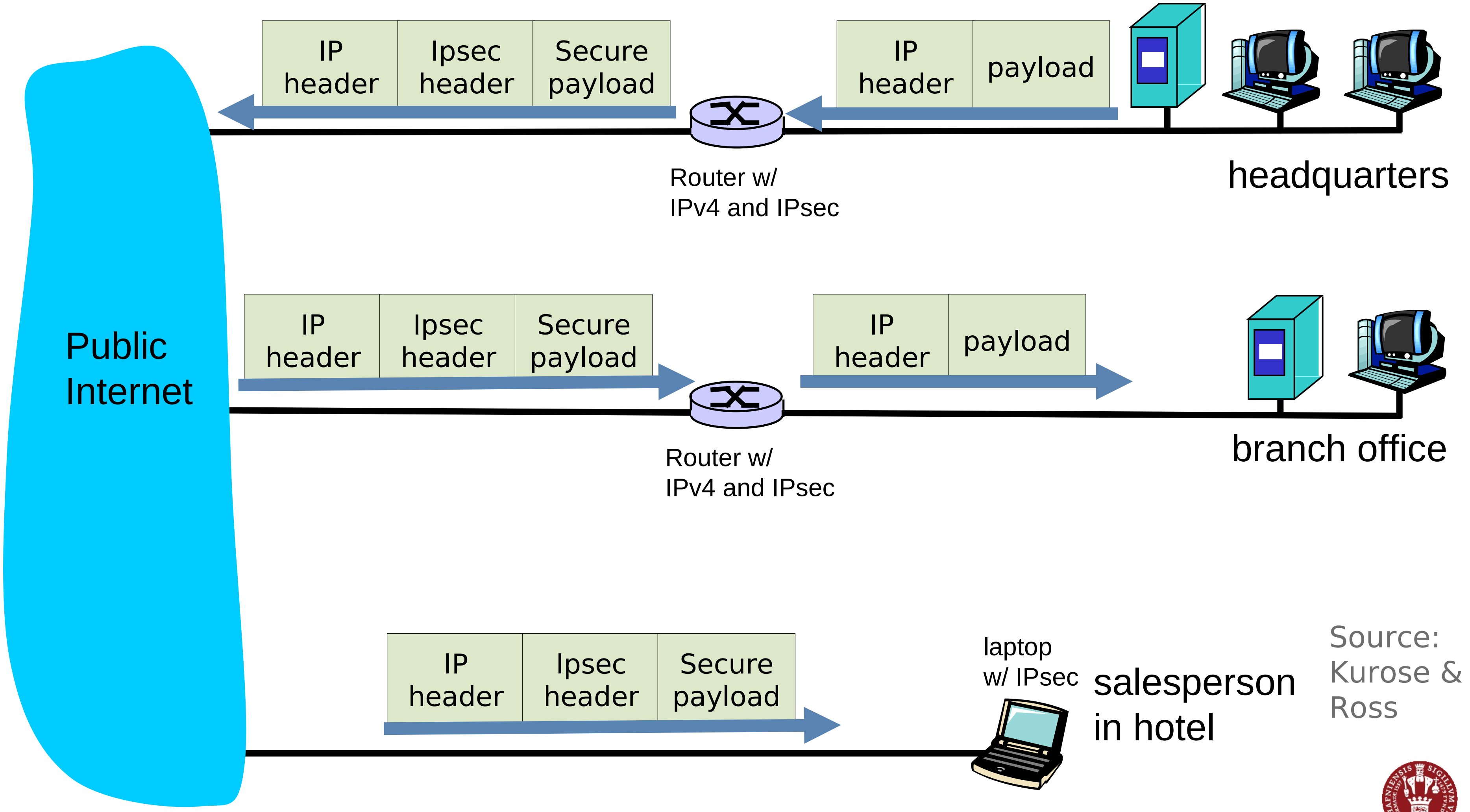
Source: Kurose & Ross

Virtual Private Network (VPN)



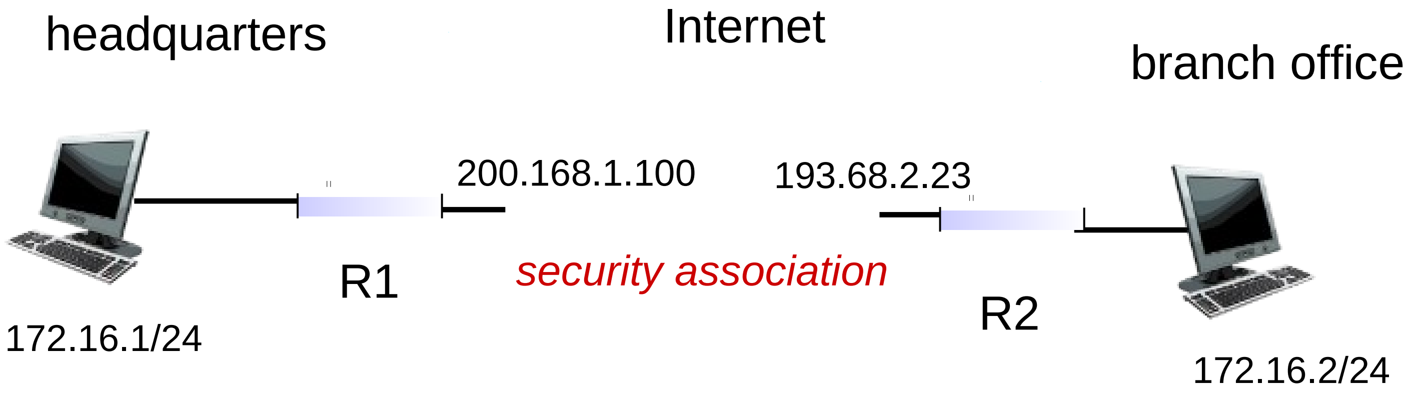
Source: Kurose & Ross

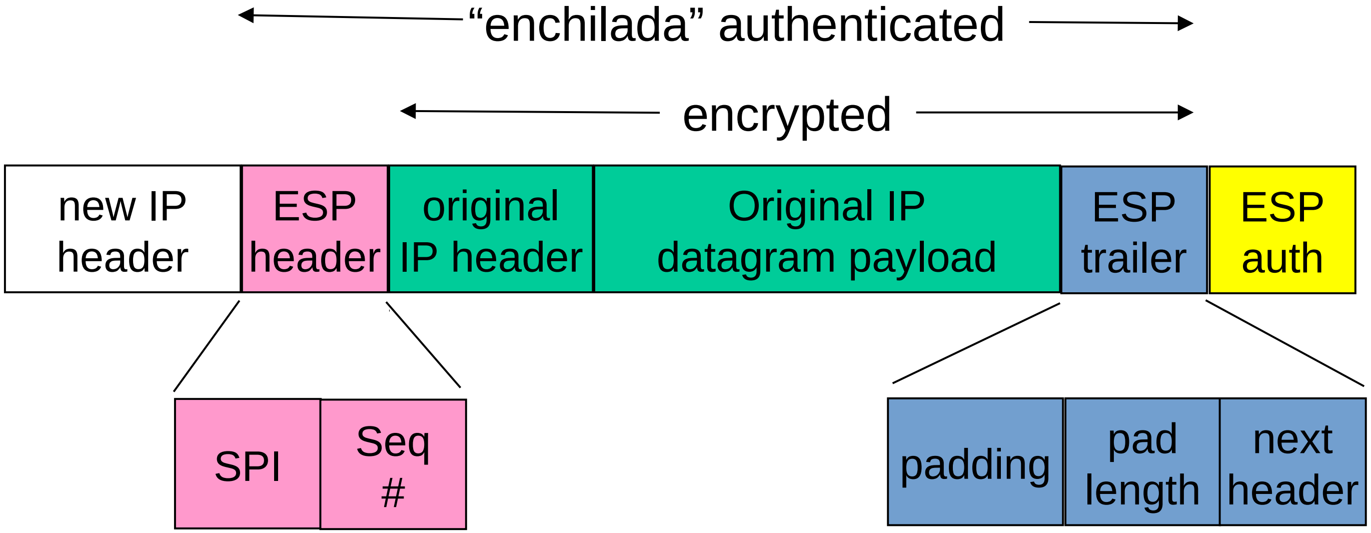
Virtual Private Network (VPN)



Source: Kurose & Ross

IPSec (What happens)





Source: Kurose & Ross

IP Security Architecture

- Specification quite complex (incl. RFC 2401, 2402, 2406, 2408)
  - Mandatory in IPv6, optional in IPv4
- Two security header extensions:
  - Authentication Header (AH)
    - Connectionless integrity, origin authentication
      - MAC over most header fields and packet body
    - Anti-replay protection
  - Encapsulating Security Payload (ESP)
    - These properties, plus confidentiality

Source: Freedman (partial)

Source: Freedman (partial)

Why is replay protection hard?

- Replay protection goal: Eavesdropper can't capture encrypted packet and duplicate later
  - Easy with TLS/HTTP on TCP: Reliable byte stream
  - But IP Sec at packet layer; transport may not be reliable

- IP Sec solution: Sliding window on sequence #'s
  - All IPSec packets have a 64-bit monotonic sequence number
  - Receiver keeps track of which seqno's seen before
    - [lastest – windowsize + 1 , latest] ; windowsize typically 64 packets
  - Accept packet if
    - seqno > latest (and update latest)
    - Within window but has not been seen before
  - If reliable, could just remember last, and accept iff last + 1
    - But IP packets can be reordered. Reordering could be particularly bad if QoS and low-priority.
    - Hence, some windows are 1024 packets.

Operational Security

- Ensure certain classes of attacks not possible by special rules and/or architectural decisions
- Rules often implemented in middleboxes
- Architectural decisions more holistic (e.g., how to prevent DoS attack?)
- Two instances today
  - Firewalls
  - DNS Security

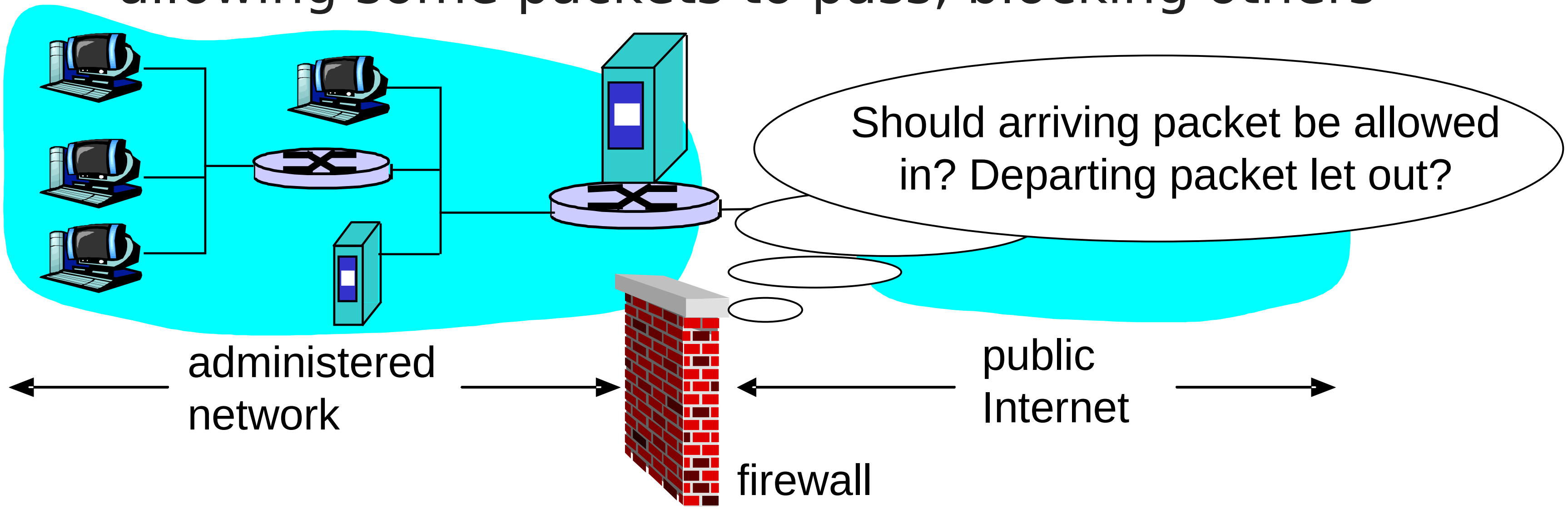
DNS Security

- Recall from lecture on DNS,
- DNS poisoning, DNS hijacking
- Addressed by randoming UDP Ids
- Also use of DNSSec protocol
- Uses public/private keys to authenticate DNS provider, as well as integrity of reply

Source: Freedman (partial)

Firewalls

- Isolates internal net from larger Internet, allowing some packets to pass, blocking others


Should arriving packet be allowed in? Departing packet let out?
administered network
public Internet
firewall

- Firewall filters packet-by-packet, based on:
  - Source/Dest IP address; Source/Dest TCP/UDP port numbers
  - TCP SYN and ACK bits; ICMP message type
  - Deep packet inspection on packet contents (DPI)

Packet Filtering Examples

- Block all packets with IP protocol field = 17 and with either source or dest port = 23.
  - All incoming and outgoing UDP flows blocked
  - All Telnet connections are blocked
- Block inbound TCP packets with SYN but no ACK
  - Prevents external clients from making TCP connections with internal clients
  - But allows internal clients to connect to outside
- Block all packets with TCP port of Quake

Source: Freedman

Source: Kurose & Ross (partial)

Configuring Firewall Rules

Policy	Firewall Setting
No outside Web access.	Drop all outgoing packets to any IP address, port 80
No incoming TCP connections, except those for institution's public Web server only.	Drop all incoming TCP SYN packets to any IP except 130.207.244.203, port 80
Prevent Web-radios (UDP traffic) from eating up the available bandwidth.	Drop all incoming UDP packets - except DNS and router broadcasts.
Prevent your network from being used for a smurf DoS attack (send ping's to targets broadcast address).	Drop all ICMP packets going to a "broadcast" address (e.g. 130.207.255.255).
Prevent your network from being tracerouted	Drop all outgoing ICMP TTL expired traffic

Clever Users Subvert Firewalls

- Example: filtering dorm access to a server
  - Firewall rule based on IP addresses of dorms
  - … and the server IP address and port number
  - Problem: users may log in to another machine
    - E.g., connect from the dorms to another host
    - … and then onward to the blocked server

- Example: filtering P2P based on port #s
  - Firewall rule based on TCP/UDP port numbers
    - E.g., allow only port 80 (e.g., Web) traffic
  - Problem: software using non-traditional ports
    - E.g., write P2P client to use port 80 instead

Source: Freedman (partial)

Honourable mentions

- Security by obfuscation
  - Use obscure/unique/outdated standards
  - Not a strategy in and of itself
- Security by location
  - Run on private networks
  - Run on hard to reach hardware

But neither of these should be considered strategies in their own right

Source: Freedman (partial)

Nothing is secure forever

“You have 1 minute to design a maze that takes 2 minutes to solve” – some scriptwriter

Source: Freedman (partial)

What should you be able to do after today?

- List security properties and related attacks
- Relate basic cryptographic schemes to their use in network protocols
- Explain the main mechanisms of HTTPS
- Explain the motivation and uses of IPSec
- Discuss operational security concerns and solutions, such as firewalls

My final lecture …

- I can supervise Bachelors projects! (#AD)
  - Contrary to what I’ve taught, I don’t do *that much* with networks
  - Main interests are in CSP based concurrency
  - Main research in event-based workflows
  - Hopefully also in privacy preserving analysis
- I will still be attending cafes whilst A4 is ongoing
- Thanks for listening :)