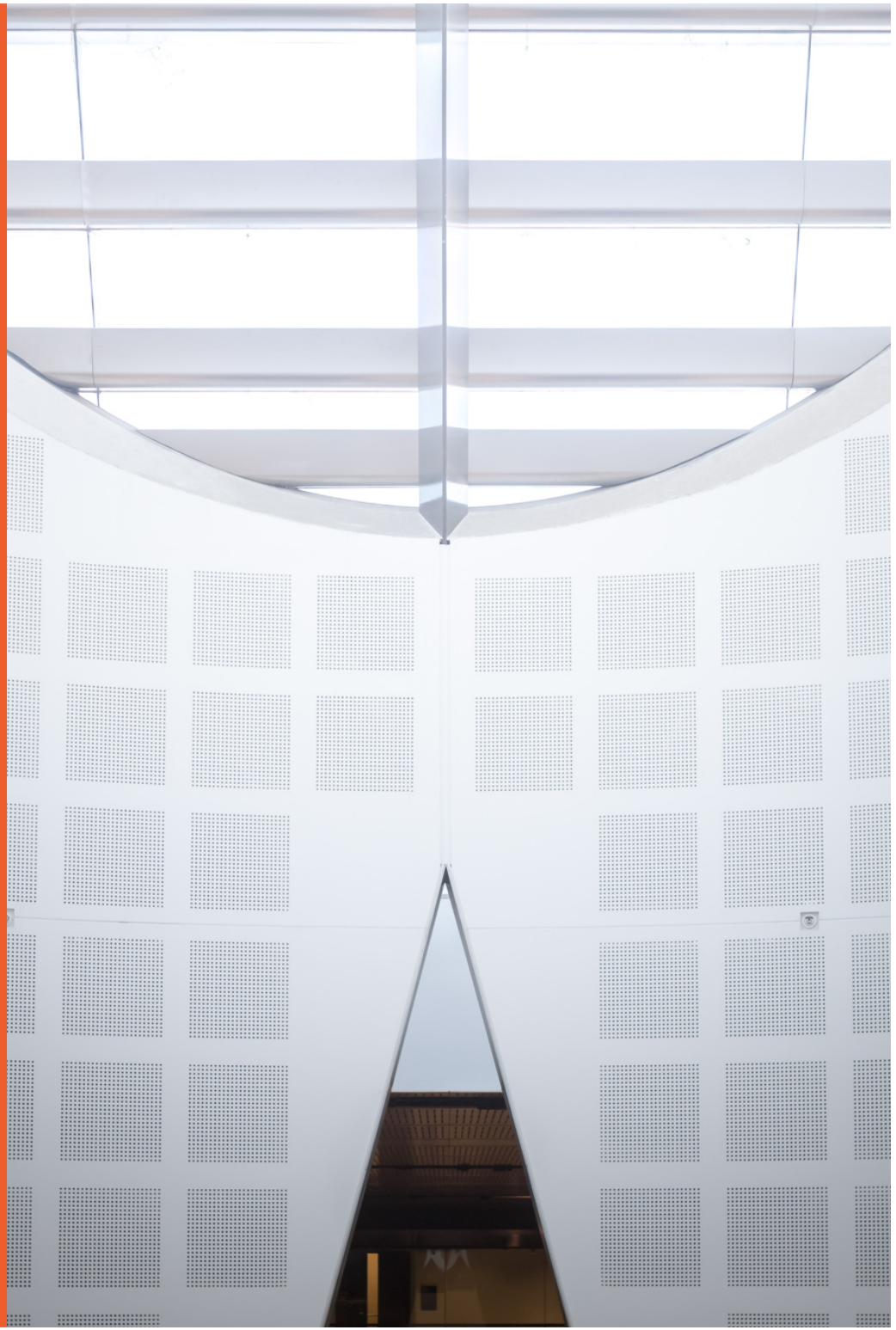
INFO 2222

Network Security

Presented by

Luke Anderson

THE UNIVERSITY OF SYDNEY



Acknowledgements

- This lecture is based to a significant extent on slides used in the unit "Network Security" at Technical University of Munich, Germany.

- The original slide deck was provided by Prof. Günter Schäfer, author of “Security in Fixed and Wireless Networks: An Introduction to Securing Data Communications”, published by Wiley.

- The slides by Prof. Schäfer have been substantially reworked by Dr. Ali Fessi, Dr. Heiko Niedermayer, Dr. Ralph Holz, Dr. Cornelius Diekmann and Prof. Georg Carle, all of TU Munich at the time of creation.

- We based our slides on the latest version of TU Munich and thank them for the permission to adap their slides.

On network security

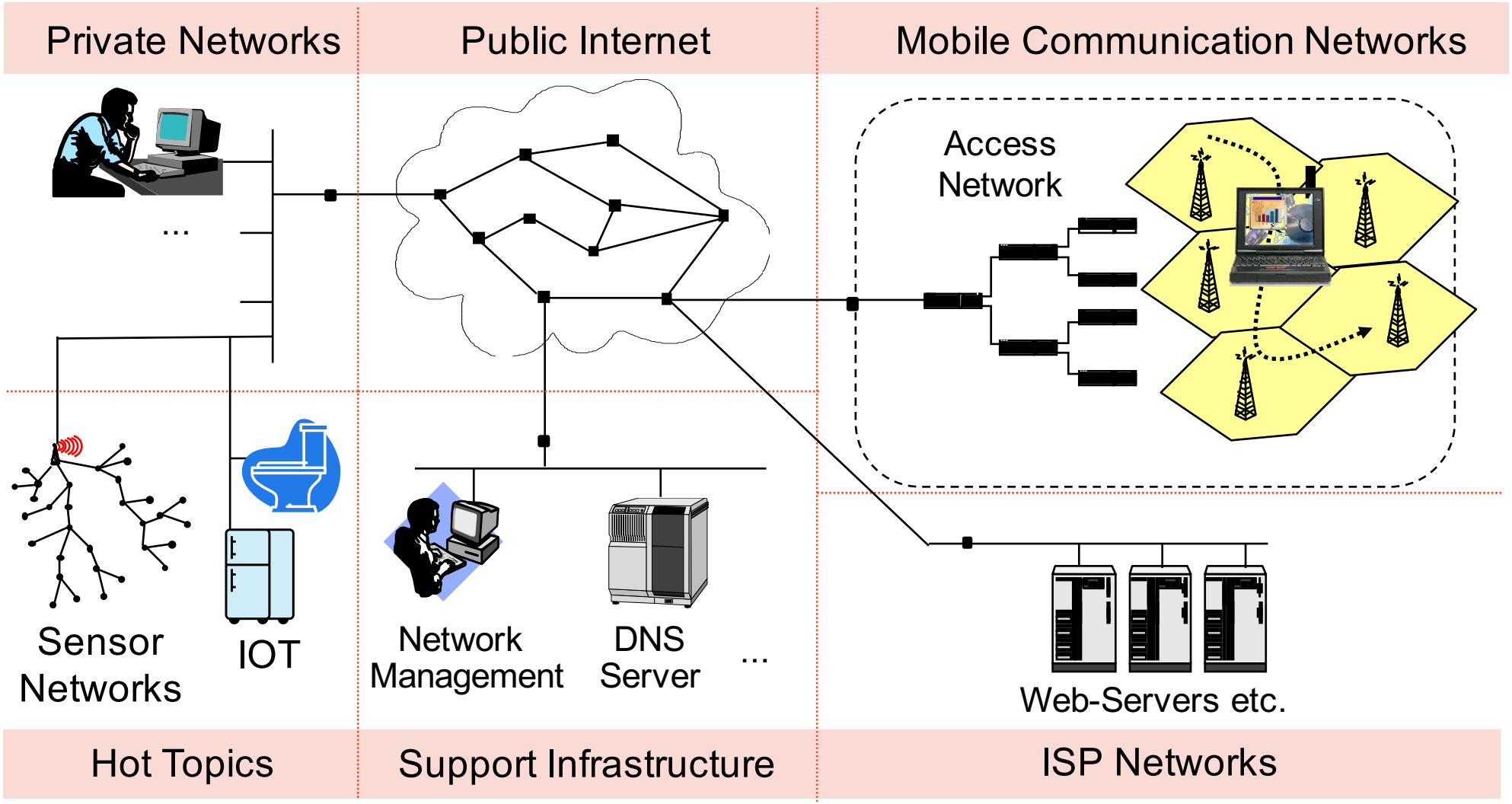
- Network security deals with attacks and defences in the context of networked systems
- The field is **huge** – we cover a very small subset here
- Agenda:
  - Motivation
  - Network security protocols
    - Network layers
    - Common protocols for authentication, confidentiality, integrity
  - Network attacks and defence
    - Denial-of-service
    - Firewalls

Overview of network security

Motivation: a changing world

- Mobile networks and ubiquitous availability of the Internet have already changed dramatically the way we
  - communicate,
  - conduct business, and
  - organize our society
- In fact, we **depend** on the networks supporting our communication
  - Business
  - Government
  - Private sphere
- The benefits associated with information and communication technology come with new ways to attack us

Attack surface in the fully networked world



What is an attack in a communication network?

- An event or sequence of actions that might lead to a violation of one or more **security goals**
  - Decisive difference to "simple" software security: the input to our programs (e.g. web server, mail server, ...) comes from remote, and there is no a priori control who can send us input
  - Further attacks become possible simply by the remote side being able to choose any kind or any amount of traffic to our own network
  - Successful intrusion into our network gives attacker plenty of opportunity to explore, breach, and exploit
- Examples of attacks (high-level):
  - An attacker breaking into a corporate computer
  - Attacker temporarily shutting down a website
  - Disclosure of emails in transit
  - Someone using services or ordering goods in the name of others
  - ...

Contextual view

- Banking:
  - Protect against fraudulent or accidental modification of transactions
  - Identify retail transaction customers
  - Protect PINs from disclosure
  - Ensure customer privacy
- Electronic trading:
  - Assure integrity of transactions
  - Protect corporate privacy
  - Provide legally binding electronic signatures on transactions
- Government:
  - Protect against disclosure of sensitive information
  - Provide electronic signatures on government documents

Contextual view

- Public telecommunication providers:
  - Restrict access to administrative functions to authorized personnel
  - Protect against service interruptions
  - Protect subscribers privacy
- Corporate / private networks:
  - Protect corporate / individual privacy
  - Ensure message authenticity
- All networks:
  - Prevent outside penetrations (who wants attackers?)

Goals in network security are familiar

- **Confidentiality**:
  - Data transmitted or stored should only be revealed to the intended audience
- **Data integrity:**
  - It should be possible to detect any modification of data
- **Accountability:**
  - Can identify the entity responsible for any communication event
  - Accountability directly supports non-repudiation, and also deterrence, intrusion prevention, security monitoring, and others
- **Availability:**
  - Services should be available and function correctly
- **Access control:**
  - Only authorized entities should be able to access certain services or information

Ways to compromise security goals in networks

- **Masquerade:**
  - An entity claims to be another entity (also called "Impersonation")
- **Eavesdropping:**
  - An entity reads information it is not intended to read
- **Loss or modification of (transmitted) information:**
  - Data is being altered or destroyed
- **Forgery of information**
  - an entity creates new information in the name of another entity
- **Denial of accountability:**
  - An entity falsely denies its participation in a communication act
- **Sabotage/Denial of Service**
  - Any action that aims to reduce the availability and / or correct functioning of services or systems
- **Authorization Violation:**
  - An entity uses a service or resources it is not intended to use

What can be broken how?

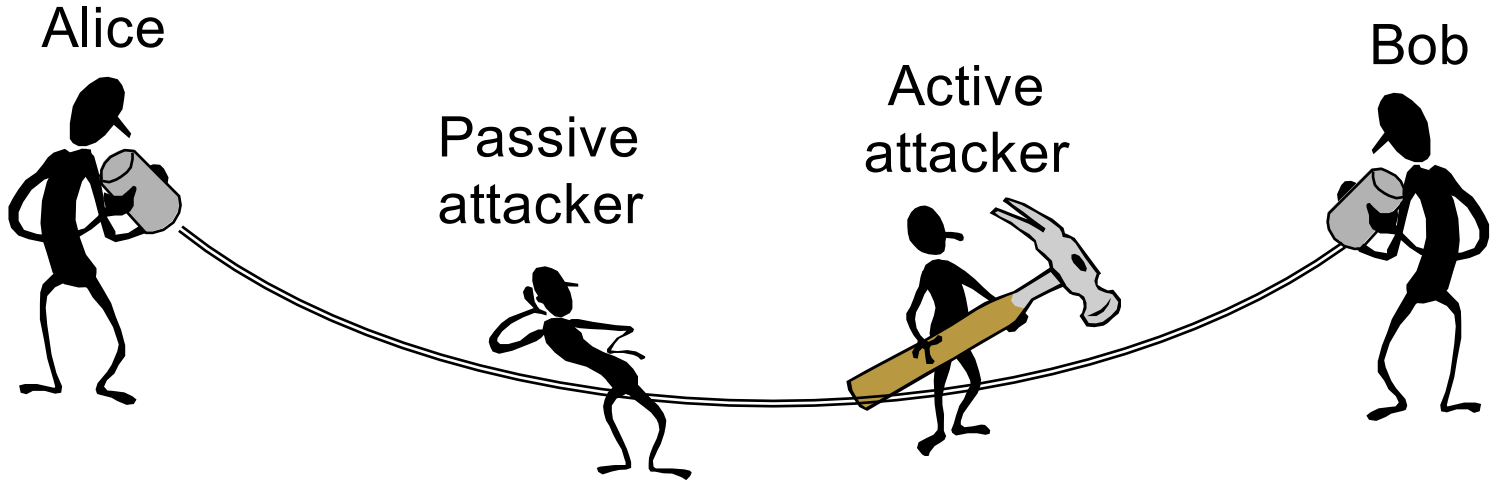
Goal	Attack taking the form of ...						
	Masquerade	Eaves-dropping	Authorization violation	Loss or modification of (transmitted) information	Denial of account-ability	Forgery of information	Sabotage/ Denial of Service
Confidentiality	X	X	X				
Integrity	X		X	X		X	
Accountability	X		X	X		X	
Availability	X		X	X	X		X
Access control	X		X		X	X	

Network security analysis

- In order to take appropriate countermeasures against threats, these have to be evaluated for a given network configuration.
- Therefore, a detailed **network security analysis** is needed that:
  - evaluates the potential risk of the threats to the entities using a network, and estimates the expenditure (resources, time, etc.) needed to perform known attacks.
  - Problem: new attacks are invented all the time;
    we cannot predict them well → drawback for defender

- Detailed security analysis of a given network configuration/ a specific protocol architecture:
  - may be required to convince financially controlling entities in an enterprise to grant funding for security enhancements
  - can be structured according to the more fine grained *attacks on the message level*.

Attacks on Communication Networks



Eavesdropping of messages/
Recognition of communication parties

Replay/delay messages

Masquerade

Message forging

Passive attacks

Active attacks

Traffic analysis

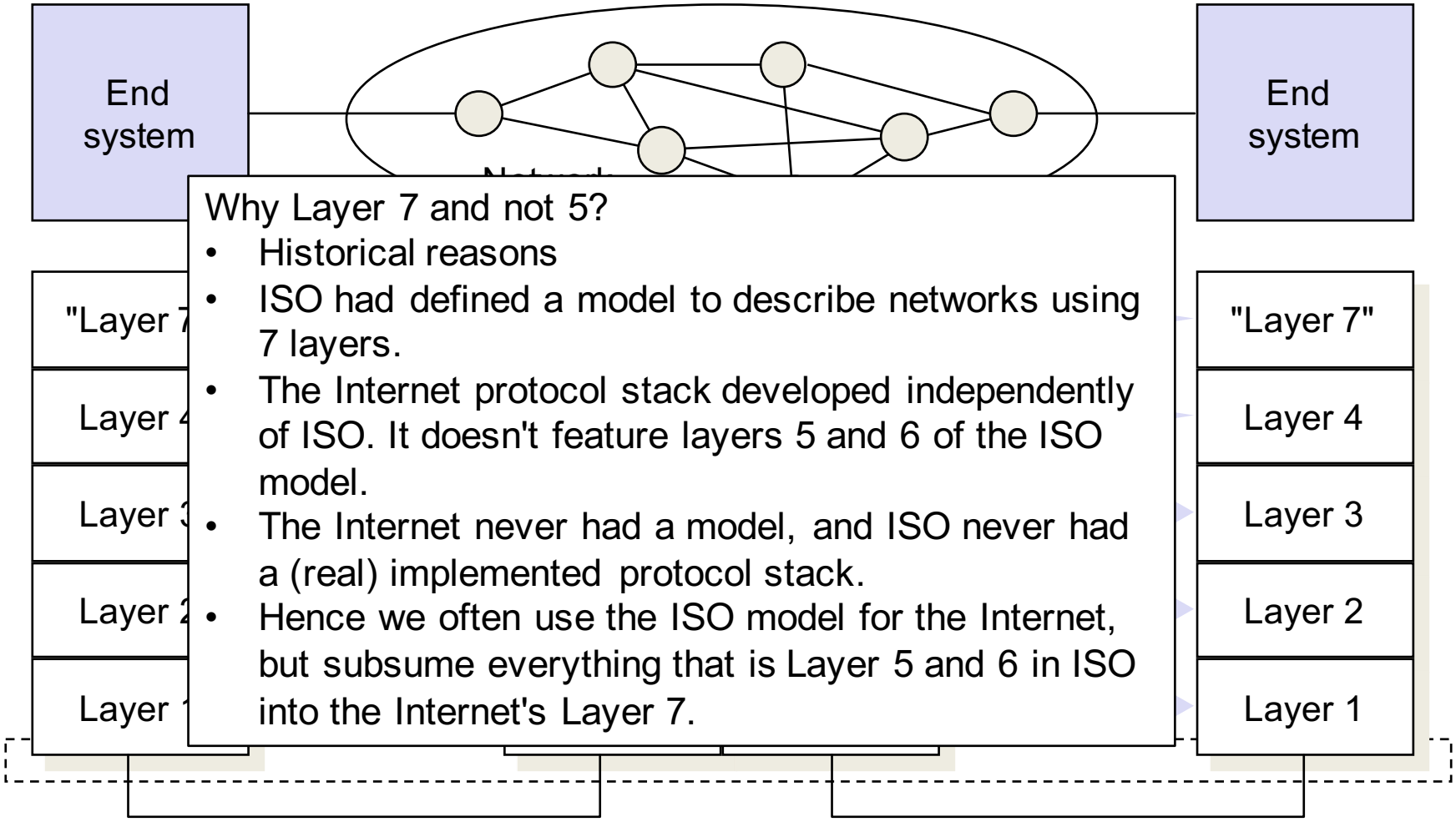
Insert / delete messages

Sabotage („Denial of Service“ attacks)

Attacking communications on the message level

- Passive attacks:
  - Eavesdropping of messages
- Active attacks:
  - Delay of messages
  - Replay of messages
  - Deletion of messages
  - Modification of messages
  - Insertion of messages
- A security analysis of a protocol architecture has to analyse these attacks according to the architecture's layers

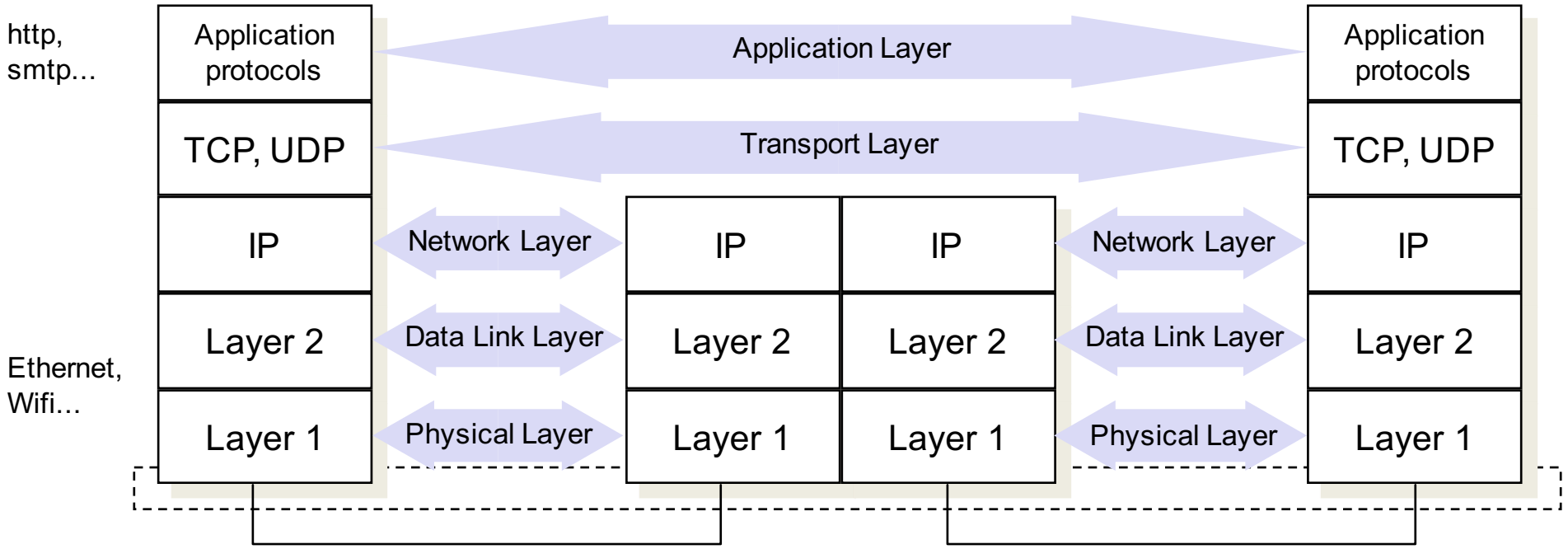
The Internet protocol stack (using ISO layers)



Why Layer 7 and not 5?

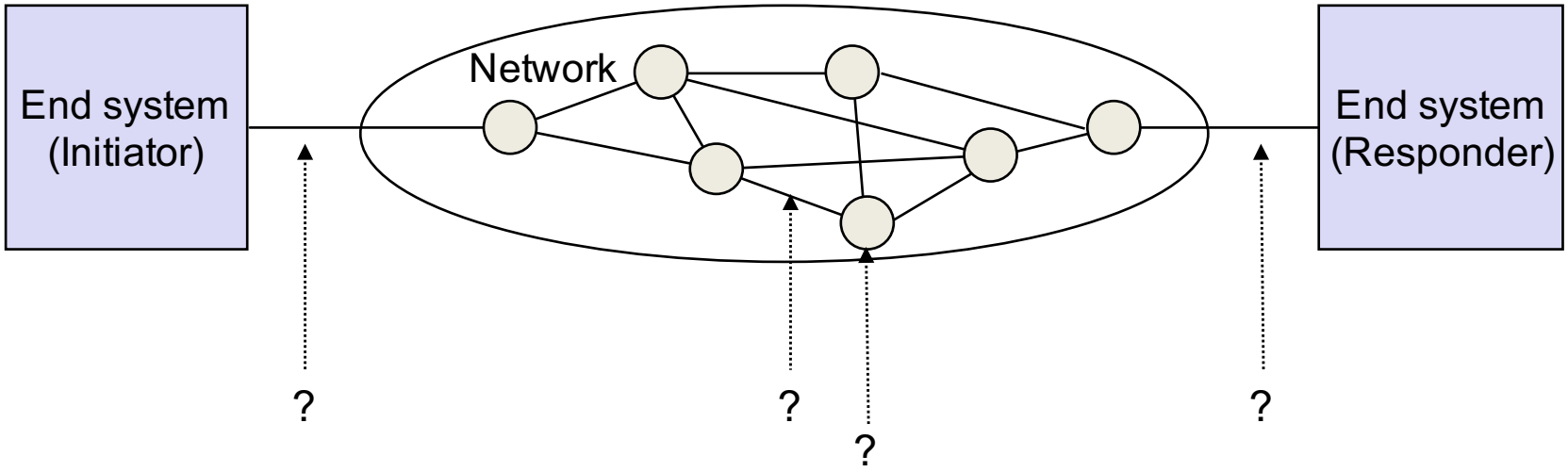
- Historical reasons
- ISO had defined a model to describe networks using 7 layers.
- The Internet protocol stack developed independently of ISO. It doesn't feature layers 5 and 6 of the ISO model.
- The Internet never had a model, and ISO never had a (real) implemented protocol stack.
- Hence we often use the ISO model for the Internet, but subsume everything that is Layer 5 and 6 in ISO into the Internet's Layer 7.

Common protocol stack today



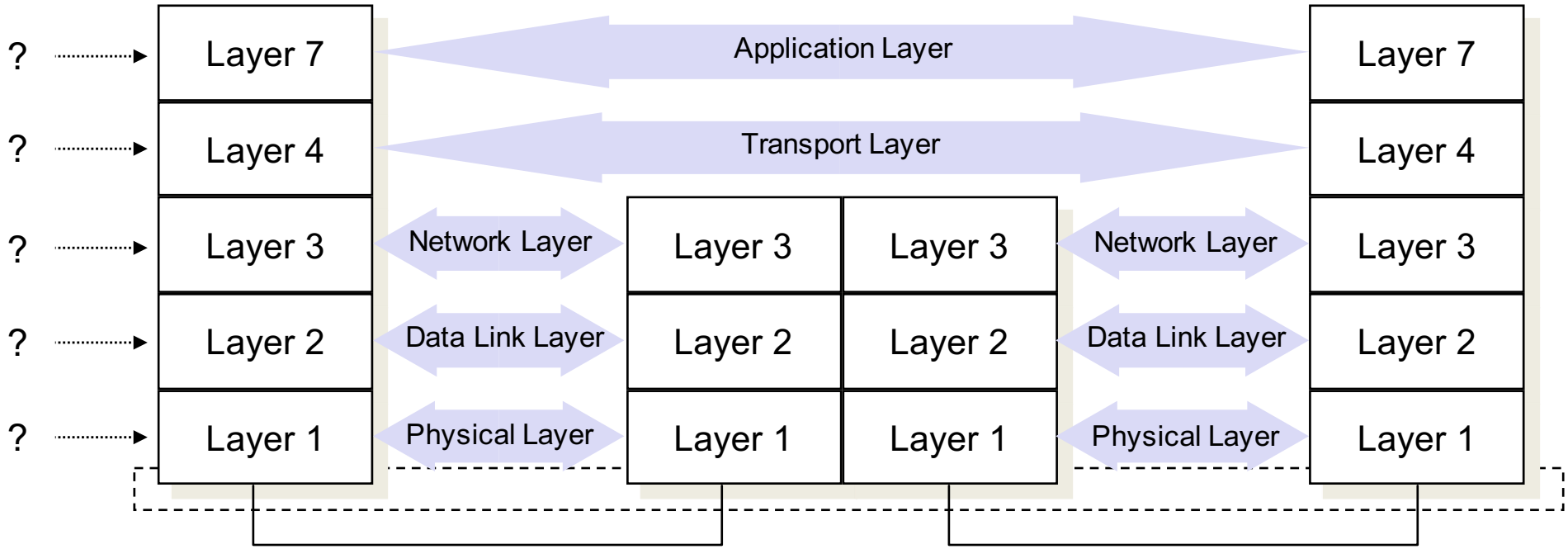
Note: in practice, standards like Ethernet and IEEE 802.11 (WiFi) define functionality of both Layer and 2 Layer 1 and do not distinguish strictly.

Network security analysis



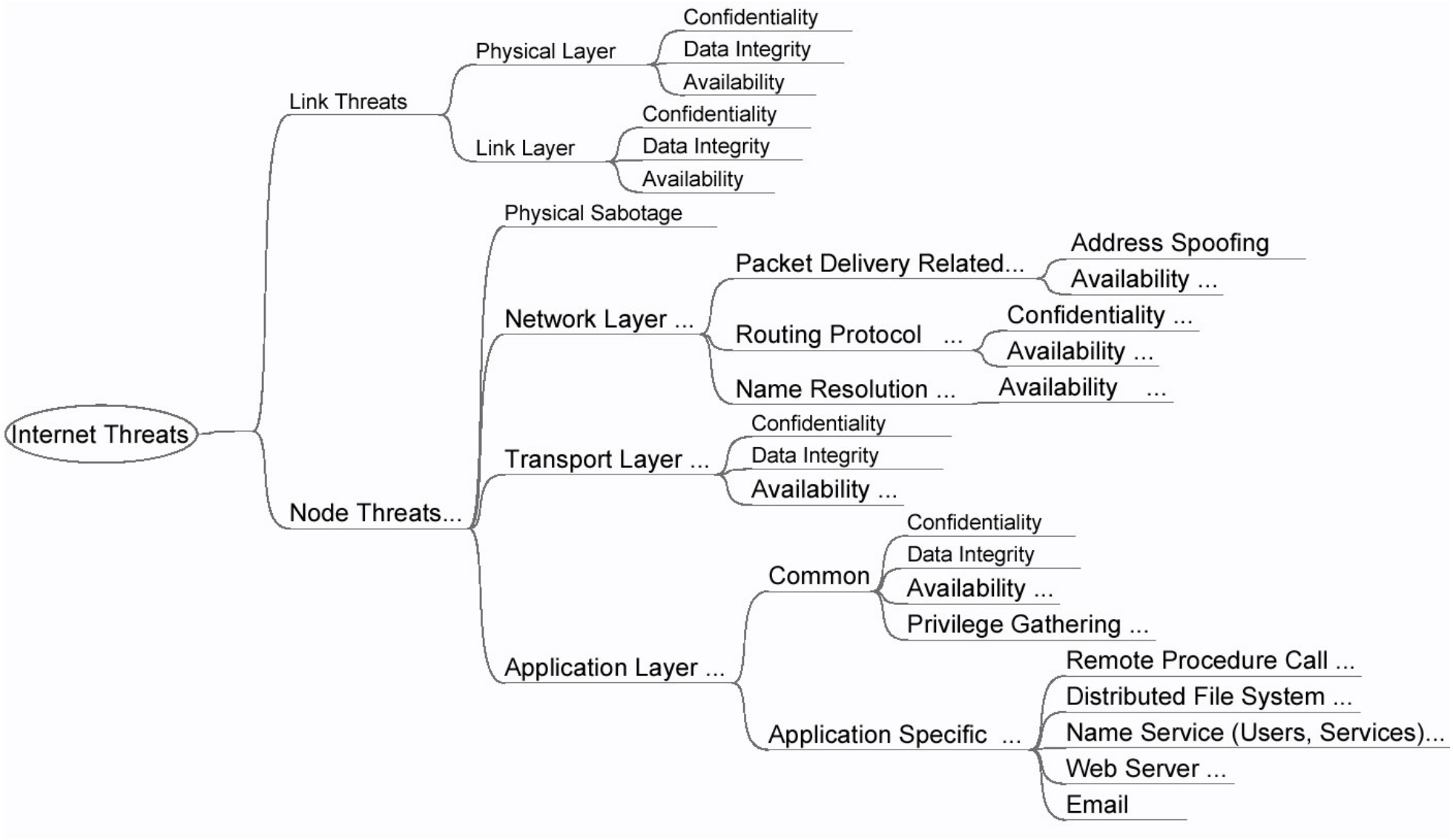
Dimension 1: At which point does the attack take place?

Network security analysis



Dimension 2: at which layer does the attack take place?

Highly complex – tree view demonstrates this



Cryptographic protocols

Example: authentication

I am Bob.

How do I prove that in the offline world?

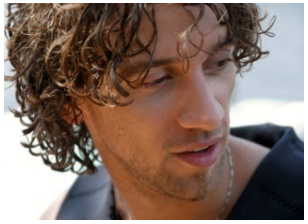
We construct passports, ID cards, etc.

My name is Bob.

Identifying the user: names & photos

My **name** is Bob. My **surname** is Surferdude. That is my **photo**.

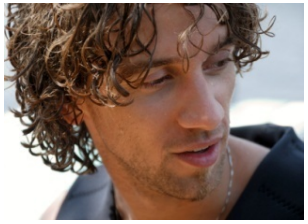
Bob S.



Signatures serve to authenticate the person

I am Bob Surferdude. Online I can **produce** such a signature.

Bob S.

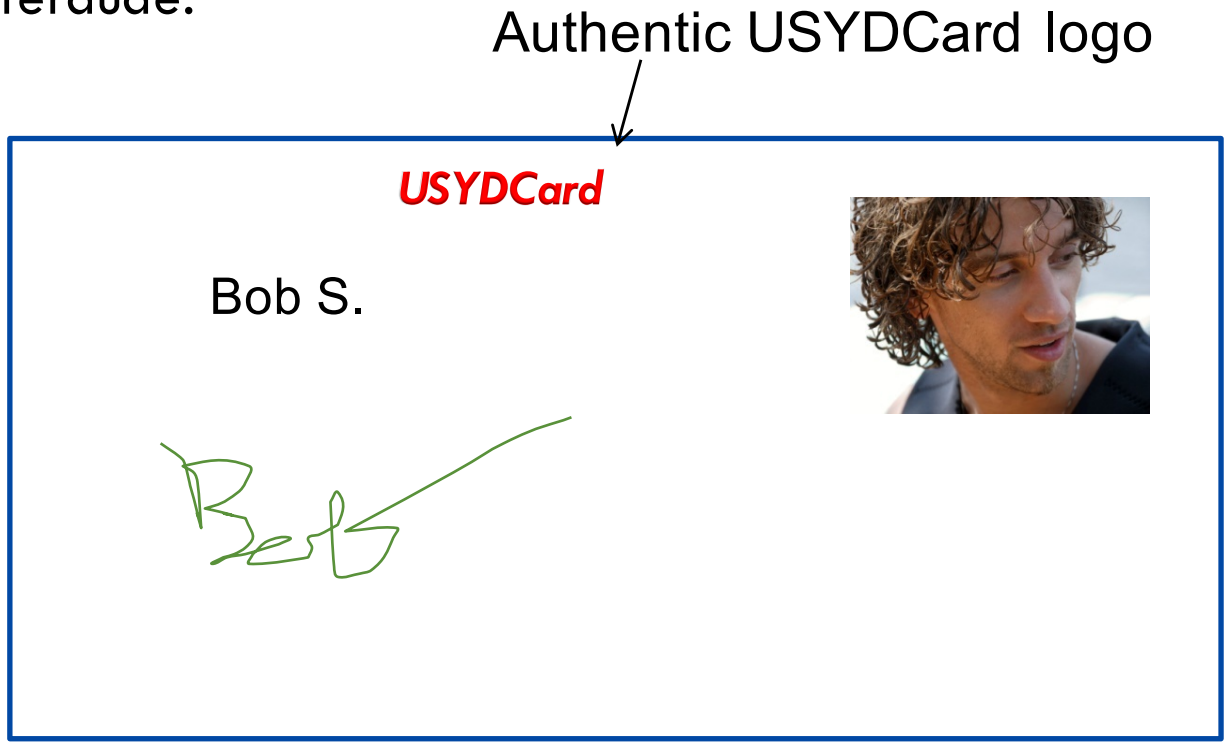


Authorities issue the card.

I am Bob Surferdude.

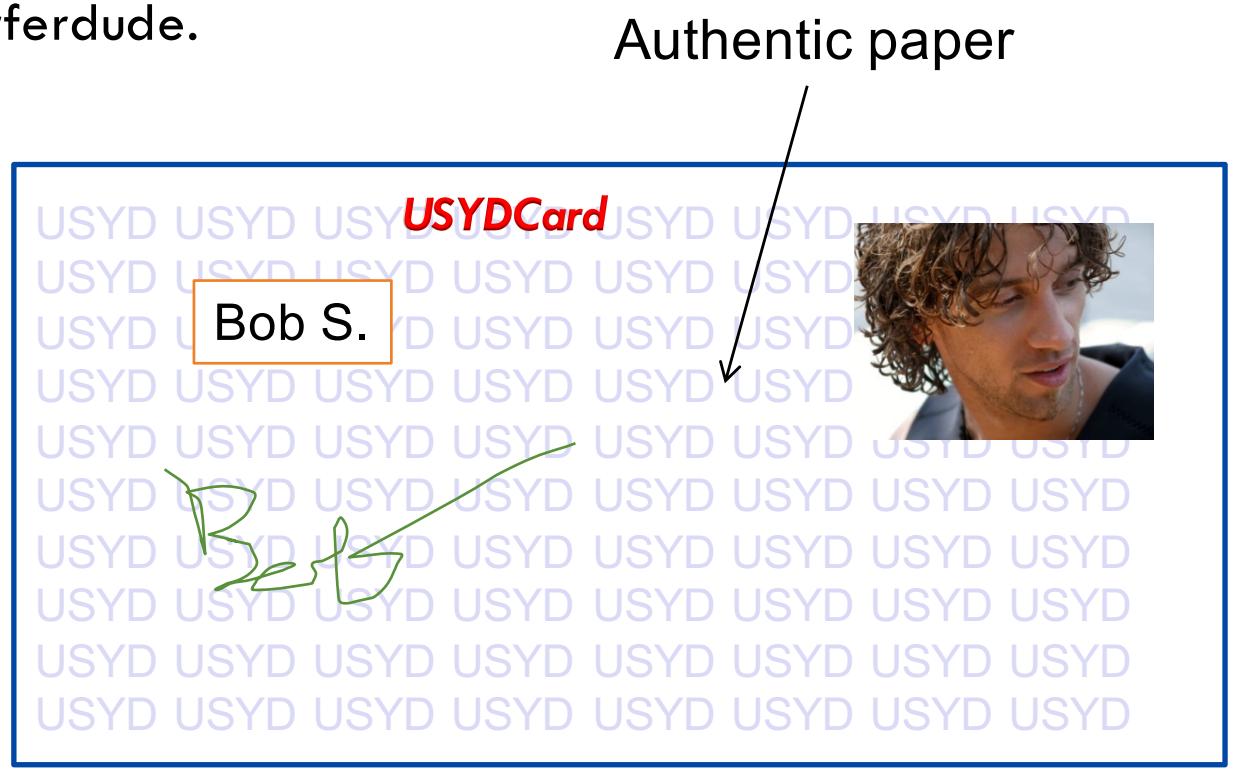
We recognize the issuing authority by some feature.

I am Bob Surferdude.



Feature is hard to forge

I am Bob Surferdude.



Authorities certify

I am Bob Surferdude.

Unforgeable signature & stamp that only USYD can generate.
You trust USYD.

How do we achieve this in the online world?

- Bob wants to:
  - Authenticate to another, online party, namely Alice
  - Communicate with Alice without another entity knowing what is being said
  - Avoid another entity being able to modify messages (without that being obvious to Bob and Alice)
- **In the online world, we use cryptographic protocols**
  - A series of steps and message exchanges between multiple entities in order to achieve a specific security objective
  - Based on the use of cryptographic primitives:
    Encryption, hash functions, signatures/MACs
  - Cryptographic protocols commonly focus on **authentication**, **confidentiality**, **integrity**
  - Can sometimes support or even enforce other goals as well

Common goals of cryptographic protocols

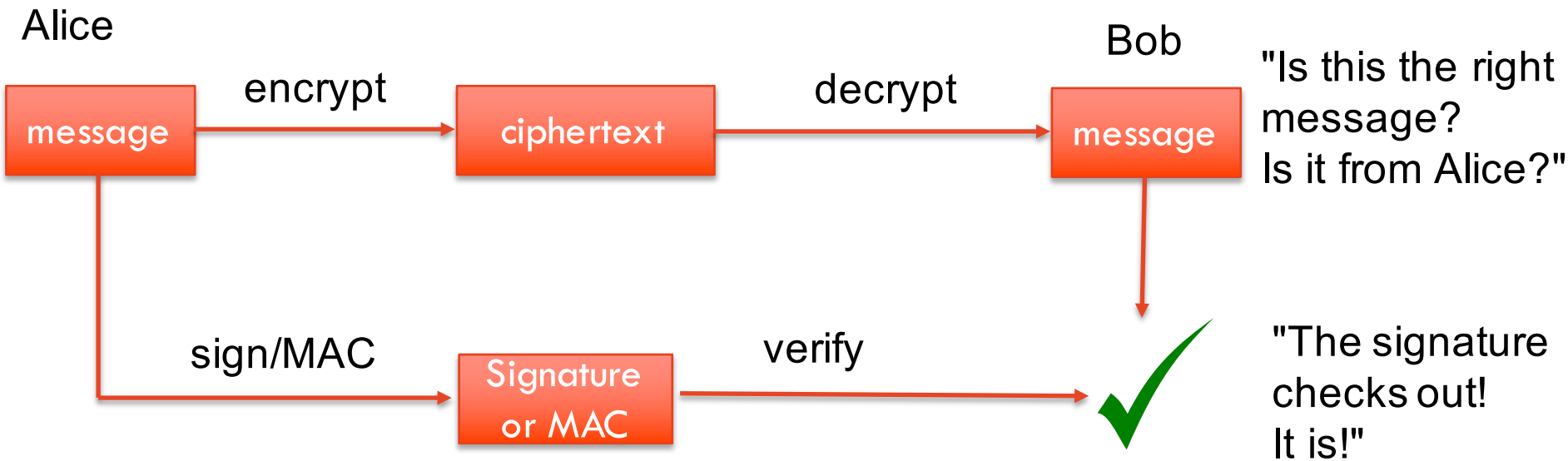
- **Confidentiality**
  - Establish secret key to encrypt every message with
  - **We use encryption: symmetric + asymmetric encryption ciphers**
- **Identity authentication:**
  - Authenticate one or both sides of communication
  - We use **signatures**
- **Message authentication:**
  - Originator of every message can be ascertained
  - We use **signatures or MACs**
- **Message integrity:**
  - Protect messages against unnoticed modifications
  - We use **MACs** (rarely: signatures)

Recap: key establishment

- Symmetric cryptography
  - Much faster than asymmetric cryptography
  - Hence we use it to exchange or derive symmetric keys
- We already know the options for key establishment
  - One side creates a secret and sends it RSA-encrypted & signed
  - Or we use **Diffie-Hellman**, where both sides collaborate to create a shared secret that is never sent over the channel
- Today, all modern protocols prefer **Diffie-Hellman**
  - Involves both sides in key creation – no risk of one side accidentally choosing a weak secret
  - Also has a property called "Perfect Forward Secure": once the communication is completed and secrets destroyed, the attacker will never be able to learn the encryption key (can only brute-force)
  - We discuss such crypto in INFO3616

Recap: authenticity and integrity

How to guarantee that a message is not modified in-flight and is from the right sender? With digital signatures and MACs.



Recap: Message Authentication Codes and Signatures

- MACs use keyed hash functions to provide integrity
  - Hash functions take input + secret as input – they are "keyed"
  - MAC is unique for every pair of input/secret – can send along with input (plaintext or ciphertext) as a checksum

- Signatures are like MACs, but they encrypt with the **private key** (and public key is used for verification)

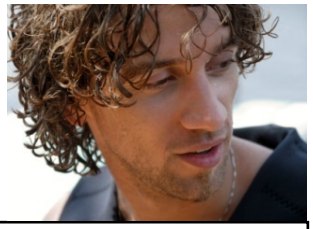
When do we use signatures and when MACs?

- MACs are faster to compute
  - Drawback: anyone who knows the secret key can create a correct MAC
  - Not a problem for participants during the protocol
  - But cannot use MAC to demonstrate to **external party** later who created some message during the protocol
- Signatures are much slower to compute
  - We tend to use them only when we have to authenticate an identity...
  - ... or if we have to prove who created some message to an external party that is not participating in the protocol

Certificates: passports of the online world

- Let's get back to our Bob example: how can Bob prove his identity?
- Certificates are used for this
- Generated by a trusted Certificate Authority (CA) for an entity
- The CA verifies the identiy out-of-band and states in the certificate that an entity and a public key correspond.
- A certificate contains
  - Cleartext
    - **Name of the entity (e.g. Bob)**
    - **Public Key of entity**
    - Name of the CA
    - (optionally) further data about the entity
    - (optionally) more data about CA
  - **Signature by the CA**
    - Hash value of cleartext signed with private key of CA

Trusted Root Certificate
--- for ----
Name: GlobalCA
Public Key:
RSA 29302048934
….
--- by ---
CA: GlobalCA
--- Signature ---
4850300434040



Certificate
--- for ----
Name: Bob S.
Public Key:
RSA 47399844398
….
--- by ---
CA: GlobalCA
--- Signature ---
10493850405

Alice, Bob, and all other entities have stored this certificate on their device because they trust this authority.
➔ They know its public key!

Public Key Infrastructures (PKIs)

- PKIs are built around one or more CAs (trusted by all)
  - The idea is that a number of CAs are allowed to issue certificates, which are recognized by all relying parties (verifiers)
  - Each entity has a public key/private key pair
  - CAs issue certificate that binds „entity name“ to public key
  - A Certificate Authority (CA) asserts the correctness of the certificate by signing it with her private key.
  - Furthermore, each entity knows the public keys of the CAs
- When Alice wishes to communicate with Bob, she can receives Bob‘s certificate
  - E.g. from a directory service or from Bob himself
- Since Alice knows CA’s public key, she can verify the signature of Bob’s certificate that was generated by CA

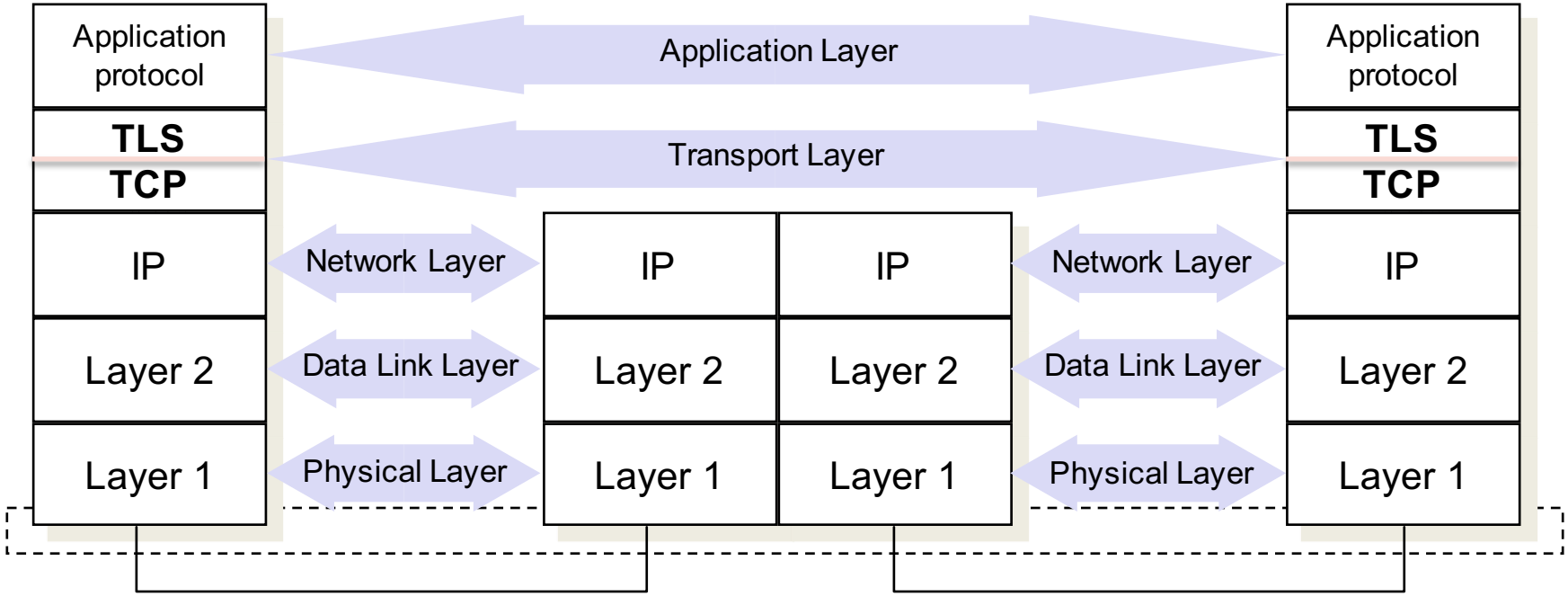
PKIs – practical limitations

- PKIs assume working, honest relationships
  - Entities must trust all CAs (else cannot verify all certificates) – not very realistic for users to **choose** them
  - Entities **must trust that CAs check the identity of an entity first** before they issue a certificate (else forgery is possible) – but they have **no means of verifying** that this is done properly
  - CAs certify millions of users – **mistakes happen**
  - CAs sometimes build chains: CA 1 certifies CA 2, which certifies CA 3, which finally certifies the actual entity. This exponentiates the problems above.
- PKIs are very commonly used (due to lack of better method):
  - "Good enough" security for most cases on the WWW
  - CAs' public keys are shipped with Internet browsers and OSes
  - I.e. every Internet browser has a list of „root CAs“ that are considered trusted

SSL/TLS: important cryptographic protocol

- **Transport Layer Security (TLS)** is the most popular security protocol on the Internet
  - Used for all WWW content and, today, much of videostreaming
  - Initial version was called Secure Sockets Layer by its inventors (in 1995), but standardized as TLS 1.0 in 1999.
  - Technically, only the TLS protocol is used today, but many people still say SSL when they mean TLS.
  - Newest version is TLS 1.3 – massive update, currently rolled out – but TLS 1.2 still most common
- If you were to use a cryptographic protocol in something you build, it would probably be TLS

Position of TLS in the network stack



TLS is technically a "Layer 4.5" protocol. The functionality it offers to the application layer is identical to TCP, except with security added. TLS creates an underlying TCP connection and sends its secured data over it.

TLS security guarantees

- TLS offers many of the security properties we know:
  - Authentication of the server (and client authentication optional) using certificates
  - Encryption
    - Diffie-Hellman key exchange (asymmetric), then continue with symmetric cipher
    - Supports all modern ciphers (ciphers state-of-the-art: ChaCha20 and AES-GCM)
    - Perfect Forward Secrecy thanks to Diffie-Hellman
  - Integrity thanks to MACs
  - Upgrade Compatibility : Client offers algorithms, server selects

TLS1.2 handshake

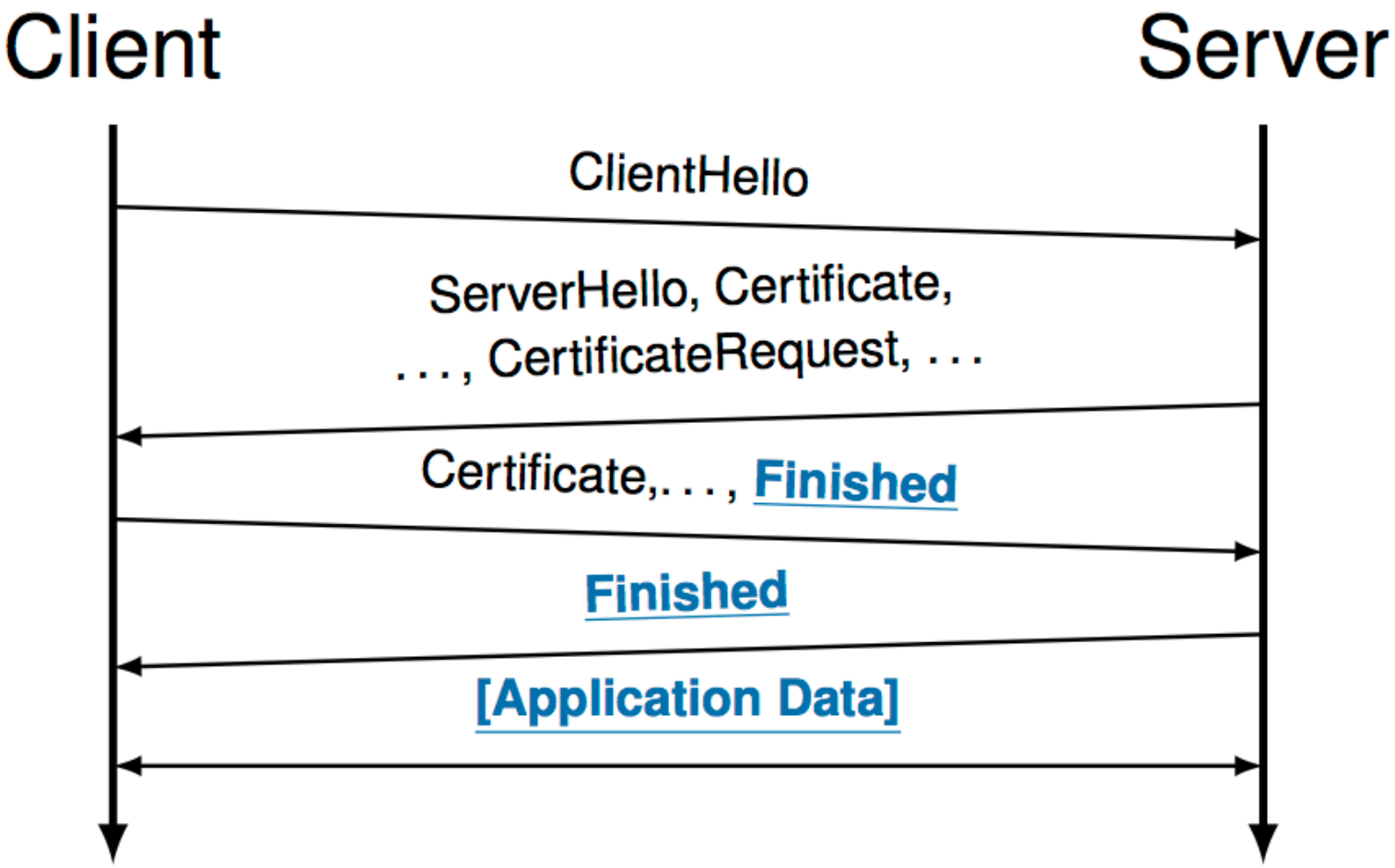


Figure: TLS 1.2 handshake, Unencrypted Data, [Encrypted Data]

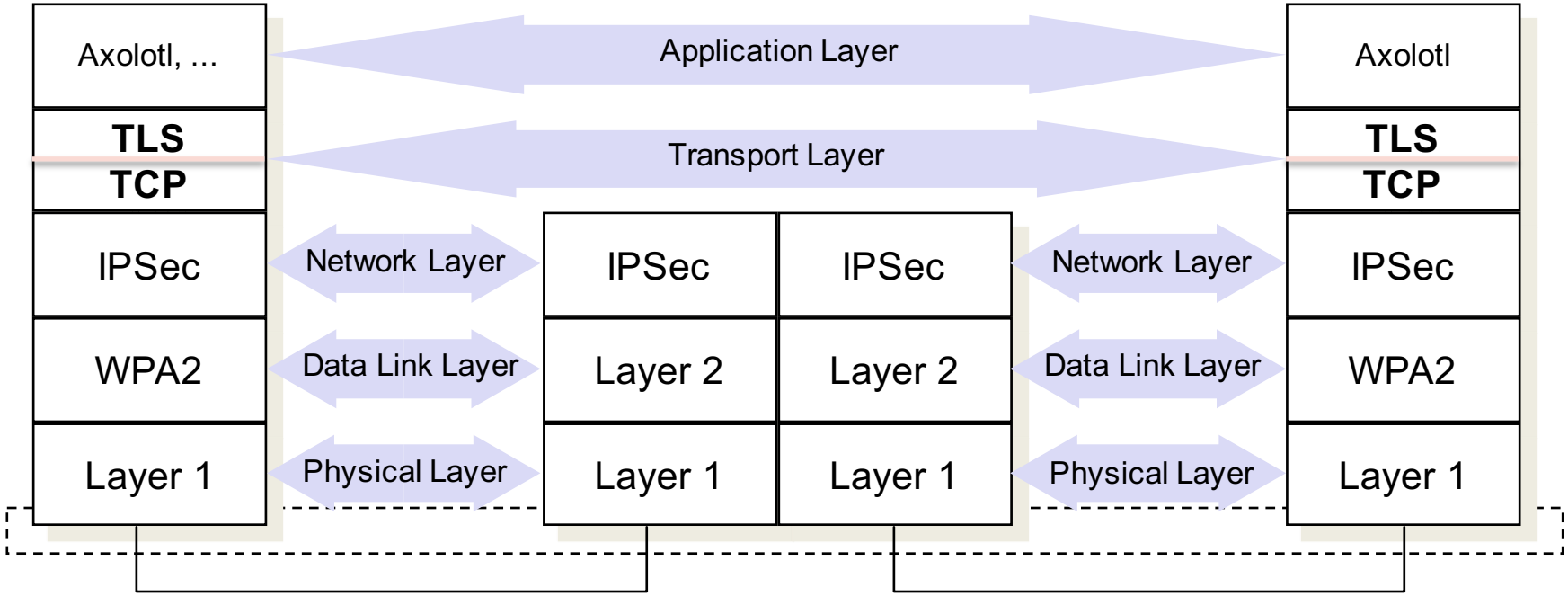
Security protocols on other layers

- Transport Layer is convenient place for a security protocol
  - But it only secures communication between **two applications**
  - Some lower layers may also require security
  - Reason 1: by far not all applications can use TLS!
  - Reason 2: can have better performance on lower layers
- Wireless networks use encryption
  - Problem: anyone can "listen in" on the air waves, and you don't know if the application uses TLS
  - Generally need authentication (known as "WiFi password")
- IP protocol can also be secured: IPSec
  - Used to connect **all** communication between two **networks**
  - Implemented in kernel, better performance than TLS
  - Impractical for ad-hoc communication: previous setup between networks required

What about application security protocols?

- Applications sometimes choose to implement their own security
  - TLS is not end-to-end: it connects applications.
  - Example: TLS can connect two instant messaging servers, which relay IMs between users. But messages would be decrypted on servers, and hence users would need to trust server operators.
- Some applications go to great length to make sure only the users's end device can obtain the plain text
  - IMs that do this: WhatsApp, Messenger, Hangouts, Signal all use the **Axolotl** protocol (WeChat does not)
- Designing security protocols requires extreme care
  - Very easy to get wrong – very few known correct protocols
  - Many attack vectors against the application itself exist (e.g. can another app read its memory?)

Common security protocols



Encryption is normally not added to Layer 1 as that is the physical signal.
However, security on Layer 1 is relevant in other aspects: think, e.g., of resilience to jamming.

Denial of Service attacks

Attacks on the network by impact

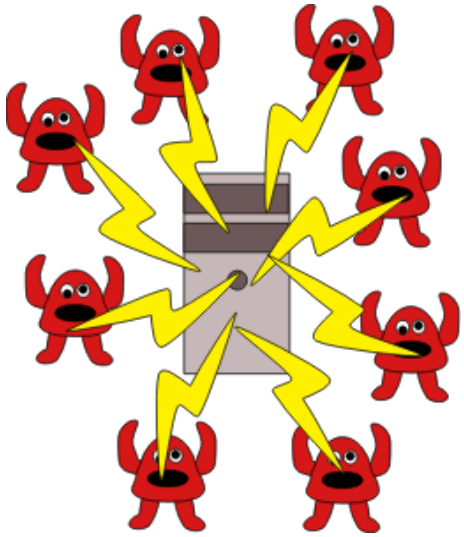
- Disruption:
  - The goal is to fully deny the victim‘s service to their clients
- Degradation:
  - Portion of the victim’s resources occupied by the attacker
  - Can remain undetected for a signification time period
  - Customers experience slow response times → move on to another Service Provider
- Data breach
  - Confidential data, passwords, password files, keys, …
- Control
  - Being able to command a machine (may not interfere with normal operation)

Denial of Service attacks

Denial of Service (DoS) attacks aim at denying or degrading legitimate users' access to a service or network resource, or at bringing down the servers offering such services

Denial of Service techniques

- *Resource destruction* (disabling services):
  - Hack into system
  - Use implementation weaknesses (buffer overflow)
  - Deviation from proper protocol execution
- *Resource depletion*:
  - Causing storage of (useless) information
  - High traffic load (high bandwidth from attacker)
  - Expensive computations ("expensive cryptography"!)
  - Resource reservations that are never used

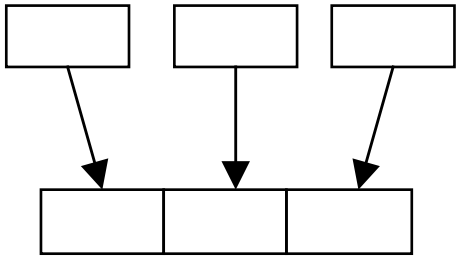


- Origin of malicious traffic:
  - Genuineness of source addresses: either genuine or forged
  - Number of sources:
    - single source, or
    - multiple sources *(Distributed DoS, dDoS)*

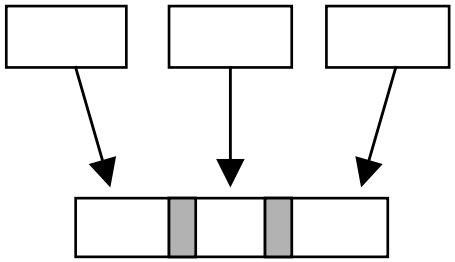
Examples: Resource Destruction (ancient)

- Ping-of-Death:
  - Maximum size of TCP/IP packet is 65536 bytes
  - Oversized packets could crash, freeze, reboot poorly programmed systems

- Teardrop:
  - Abuse of fragmentation option when splitting data into packets
  - Intentionally incorrect offsets could crash system

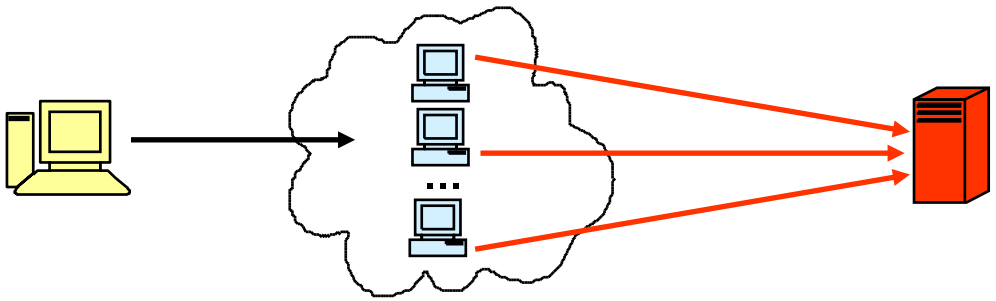


- Take-Home Message:
  - Only a few packets can be sufficient to bring down a system.



Resource depletion: abusing Layer 3

- ICMP is a protocol on Layer 3 to exchange control messages
- *Smurf* attack - ICMP broadcast:
  - An attacker sends an ICMP request to a broadcast address with the source addressed forged to refer to the victim
  - Routers (often) allow ICMP echo requests to broadcast addresses
  - All devices in the addressed network respond to the packet
  - The victim is flooded with replies to the echo request
  - With this technique, the network being abused as an (unaware) attack amplifier is also called a *reflector network:*

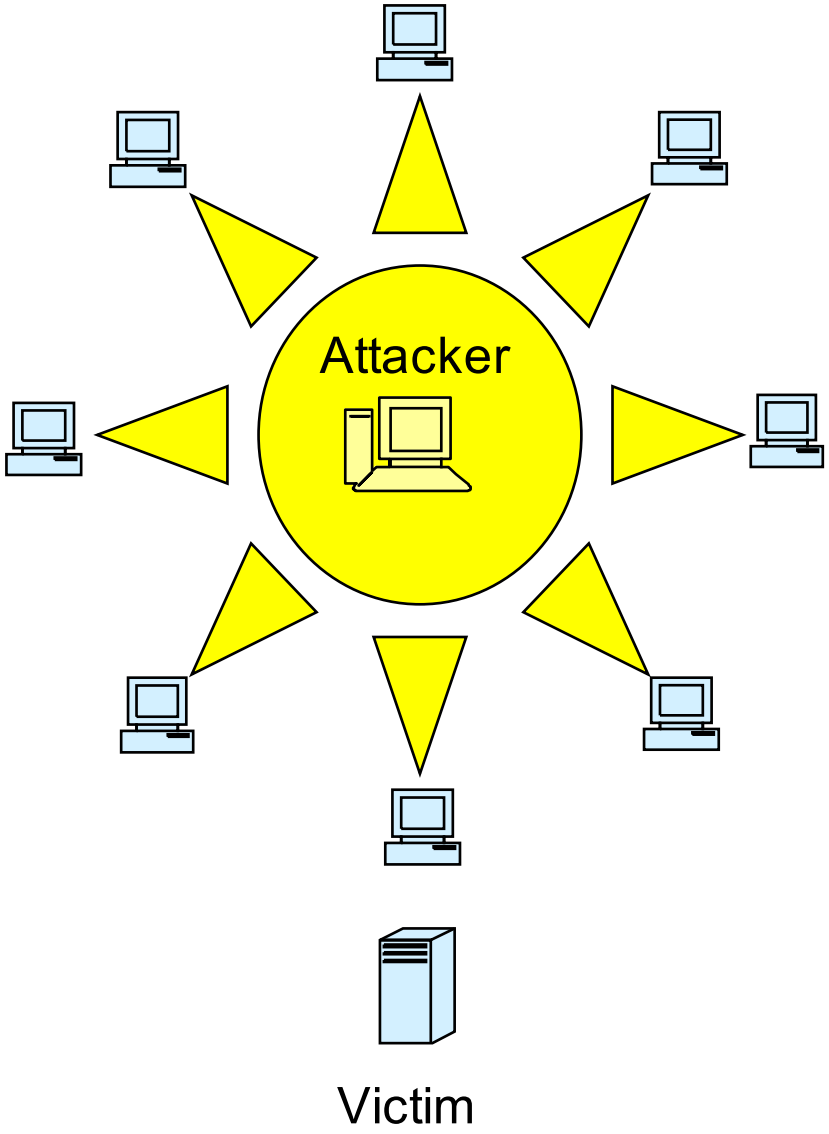


Resource depletion: abusing Layer 4

- Layer 4 wraps and transports application layer data
  - TCP starts with a handshake between the two communicating parties
  - Early TCP implementations would reserve memory when first handshake message was received (the "SYN" message)
  - Attackers could abuse that by sending a large number of handshake requests ("TCP SYN attack")
  - Sending enough SYNs would result in memory exhaustion

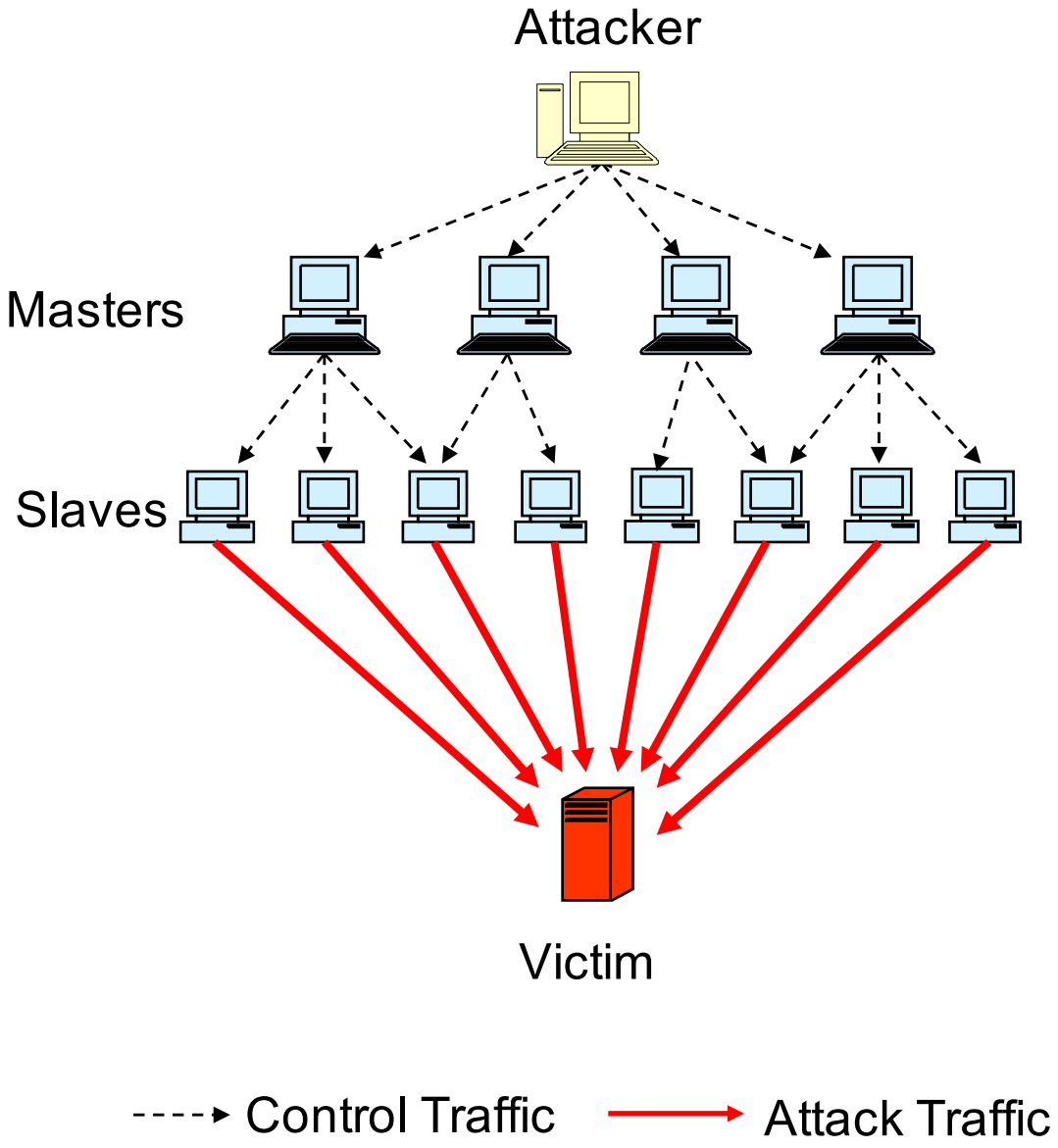
- Today's implementations are much more robust
  - Use mathematical trick to delay resource allocation until the end of the handshake

Resource Depletion with dDoS: any Layer from 3+



- Goal: overwhelm the victim with traffic
- Attacker intrudes multiple systems by exploiting known flaws
- Attacker installs DoS-software: „Root Kits“ are used to hide the existence of this software
- DoS-software is used for:
  - Exchange of control commands
  - Launching an attack
  - Coordinating the attack

Resource depletion with dDoS: alternative



- The attacker classifies the compromised systems in:
  - Master systems
  - Slave systems
- Master systems:
  - Receive command data from attacker
  - Control the slaves
- Slave systems:
  - Launch the proper attack against the victim
- During the attack there is no traffic from the attacker

Prevention: defence techniques against (d)DoS attacks

- Defences possible in many respects:
  - Defences against penetration of the network
    - Good system administration
    - Firewalls, logging & intrusion detection systems
  - Implementation defences:
    - Code reviews, testing, etc. to eliminate vulnerabilities
    - Application should remain well-performing under heavy load
  - "DoS-aware design":
    - Do not perform expensive operations, reserve memory, etc., before authentication

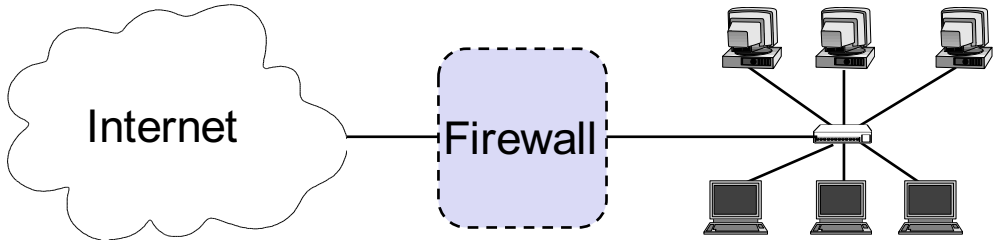
Prevention: defence techniques against (d)DoS attacks

- Defences against resource depletion:
  - Rate Control (ensures availability of other functions on same system)
    i.e. a potential reason to implement QoS mechanisms
  - Authentication of clients plays an important role for the above measures
- Concerning origin of malicious traffic:
  - Defences against single source attacks: disabling of address ranges
    (does not help if addresses are spoofed)
  - Defences against forged source addresses:
    - **Ingress filtering at ISPs**
      (block incoming packets from outside ISP with source IPs of ISP itself)
    - **Egress filtering**
      (block outgoing packets with source address from other network)
  - Widely distributed DoS: no true "solution" known!
    - Some businesses, e.g. Cloudflare, can help by providing extreme amount of clever filtering and extra resources – for a price

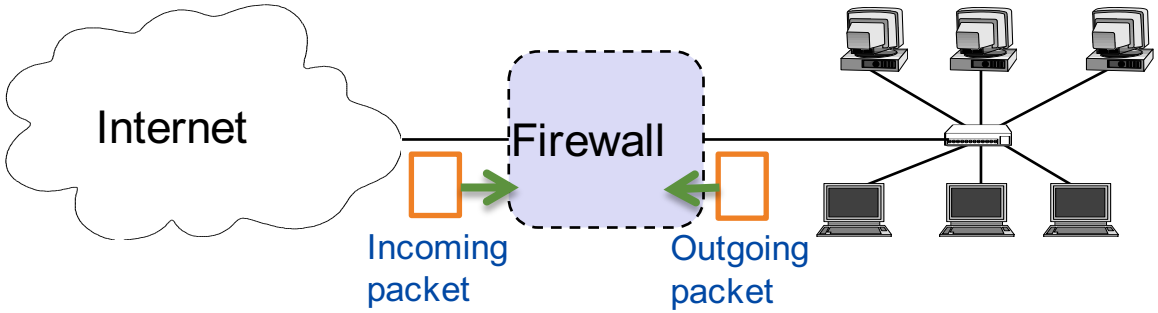
Firewalls

Introduction to network firewalls

- A network firewall acts like a gatekeeper in a fortress:
  - Gatekeeper restricts people: enter only at carefully controlled point
  - It prevents attackers from getting close to other defences
  - It restricts people to leaving at one carefully controlled point
- Firewalls realize such access control on the network level
- A network firewall is installed at a point where the protected subnetwork is connected to a less trusted network:
  - Example: Connection of a local area network to the Internet



Introduction to network firewalls



- Packets can arrive from both sides
  - *Incoming packets* from the Internet to the local network
  - *Outgoing packets* from the local network to the Internet
  - Please note:
    - The external network does not have to be the Internet, firewalls can also operate between different local networks.
    - We hence speak of external and internal networks.
- How does the firewall know what to do with the packets?
  - The firewall is configured by an administrator
  - The administrator configures rules

Basic strategies for default rules

- Default deny strategy: (~ Whitelisting)
  - *"Everything that is not explicitly permitted is denied"*
  - Examine the services the users of the protected network need
  - Consider the security implications of these services and how the services can be safely provided
  - Allow only those services that can be safely provided and for which there is a legitimate need
  - Deny any other service
- Default permit strategy: (~ Blacklisting)
  - *"Everything that is not explicitly forbidden is permitted"*
  - Permit every service that is not considered dangerous
  - Example:
    - Network file system (NFS) is not permitted across the firewall
    - Incoming SSH connections are only allowed to one specific host

What can a firewall do with a packet?

- The firewall forwards the packet.
  - This is the common operation in case the packet belongs to a flow or application that you want to allow.
  - Typical terms for this: Allow / Permit / Accept / Pass
- The firewall deletes the packet and does not forward it.
  - This is the common operation in case the packet belongs to a flow or application that you want to stop.
  - Typical terms for this: Drop / Deny / Reject
- Other options include
  - Log that a certain type of packet appeared,
  - Send error message to sender (e.g. via ICMP)
  - Rate limit, mirror, inform the admin, etc.

Information that firewalls have access to

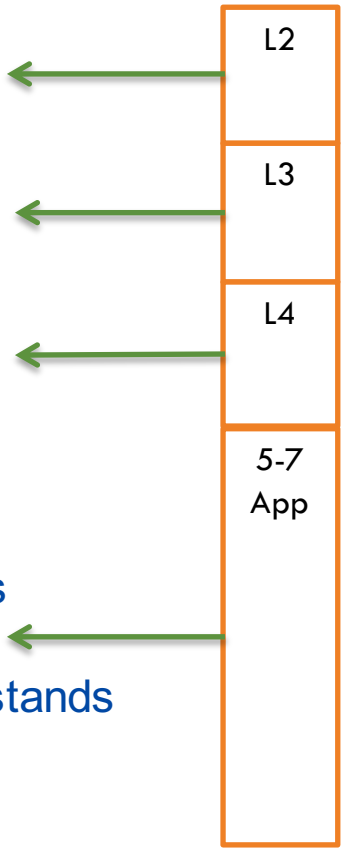
- How can a firewall gain information for its decision?
  - Without additional state-keeping, it can only use the packet and the data in its header fields.
  - It determines the entities, protocols, protocol states, and/or application to make the decision.

Link Layer: direction of packet, next physical hop

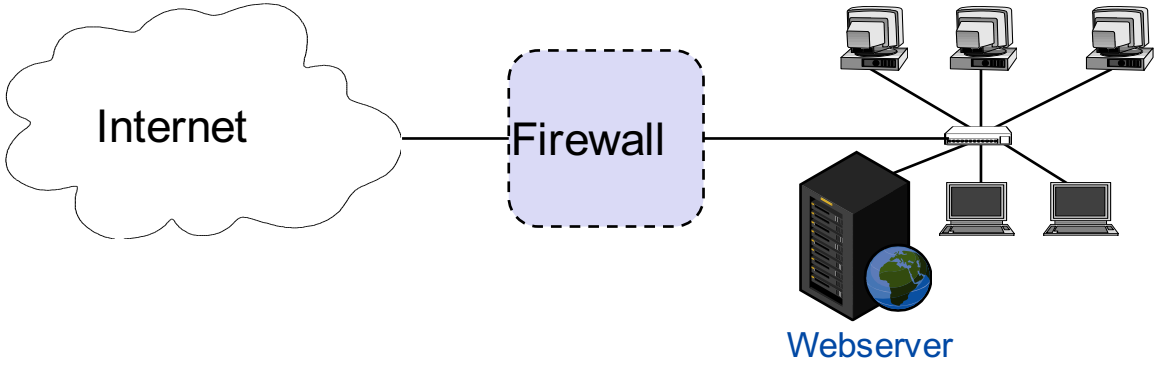
Network Layer: communication end points (entities, e.g. IPv4 addresses), transport protocol

Transport Layer: ports (applications), protocol state

Application Layer: Application Layer Filtering requires an application level packet filter. This is not standard in firewall operation. Requires that the firewall understands the application protocol.



Example scenario



- Allow HTTP traffic initiated by external hosts to webserver (TCP port 80)
- Allow internal hosts to initiate
  - HTTP traffic to Internet (TCP, port 80)
  - DNS traffic to Internet (UDP, port 53)
- Do not allow other communication, in particular no communication initiated by external hosts to the local hosts other than the webserver.

Firewall Scenario 2 – Example *Stateful* Filtering

Rule	Direction	Src. Addr.	Dest. Addr.	Protocol	Src. Port	Dest.Port	State	Action
A1	Inbound	External	Webserver	TCP	>1023	80	ANY	Permit
A2	Outbound	Websrv.	External	TCP	80	>1023	EST	Permit
B1	Outbound	Internal	External	TCP	>1023	80	NEW	Permit
B2	Inbound	External	Internal	TCP	80	> 1023	EST	Permit
C1	Outbound	Internal	External	UDP	>1023	53	NEW	Permit
C2	Inbound	External	Internal	UDP	53	>1023	EST	Permit
D	Either	Any	Any	Any	Any	Any	ANY	Deny

- Idea: track state of every connection: new, established, or "any state"
  - Decide if certain packet belongs to allowed traffic category or not
- For UDP, such firewalls use heuristics to decide whether a UDP packet belongs to a "connection" or not
- Note the Deny-All rule at the end for all non-matching traffic

Common firewall policy errors

- How is your firewall management interface reachable?
  - From the Internet? From the complete internal network?
  - Via telnet? Via UPnP?
- What is allowed over the Internet?
  - NetBIOS? NFS? RPC? Telnet? SSH?
- IPv4 and IPv6?
  - Are the rule sets compliant?
- Outbound rule *ANY*?
  - Even private IP ranges?
  - Even from IP ranges that don't belong to you?

Conclusion to network firewalls

- What firewalls can do:
  - A firewall is a focus for security decisions
  - A firewall can enforce a security policy, i.e. concerning access control
  - A firewall can log Internet activity efficiently
  - A firewall can block unwanted traffic if the traffic can be characterized,
    - e.g. with an IP 5-tuple: IP source address, IP destination address, source port number, destination port number, transport protocol
  - A firewall can limit exposure to security problems in one part of a network

- Note: we discussed only very simple firewalls here. More complex rulesets can, e.g., track connections over time. They are not more powerful in what they can do, however – but policies can be easier to define.

Conclusion to network firewalls

- What firewalls cannot do:
  - A firewall can't protect against malicious insiders
  - A firewall can't protect against connections that don't go through it
  - A firewall can't protect against completely new threats
  - A firewall can't fully protect against viruses,
    - e.g. if viruses are spread through emails, and the email service is allowed through the firewall, which is typically the case
  - A firewall does not perform cryptographic operations, e.g. message authentication
  - A firewall can't set itself up correctly ($\Rightarrow$ cost of operation)