Imperial College London

Computer Networks and Distributed Systems

Security

Course 527 – Spring Term 2014-2015

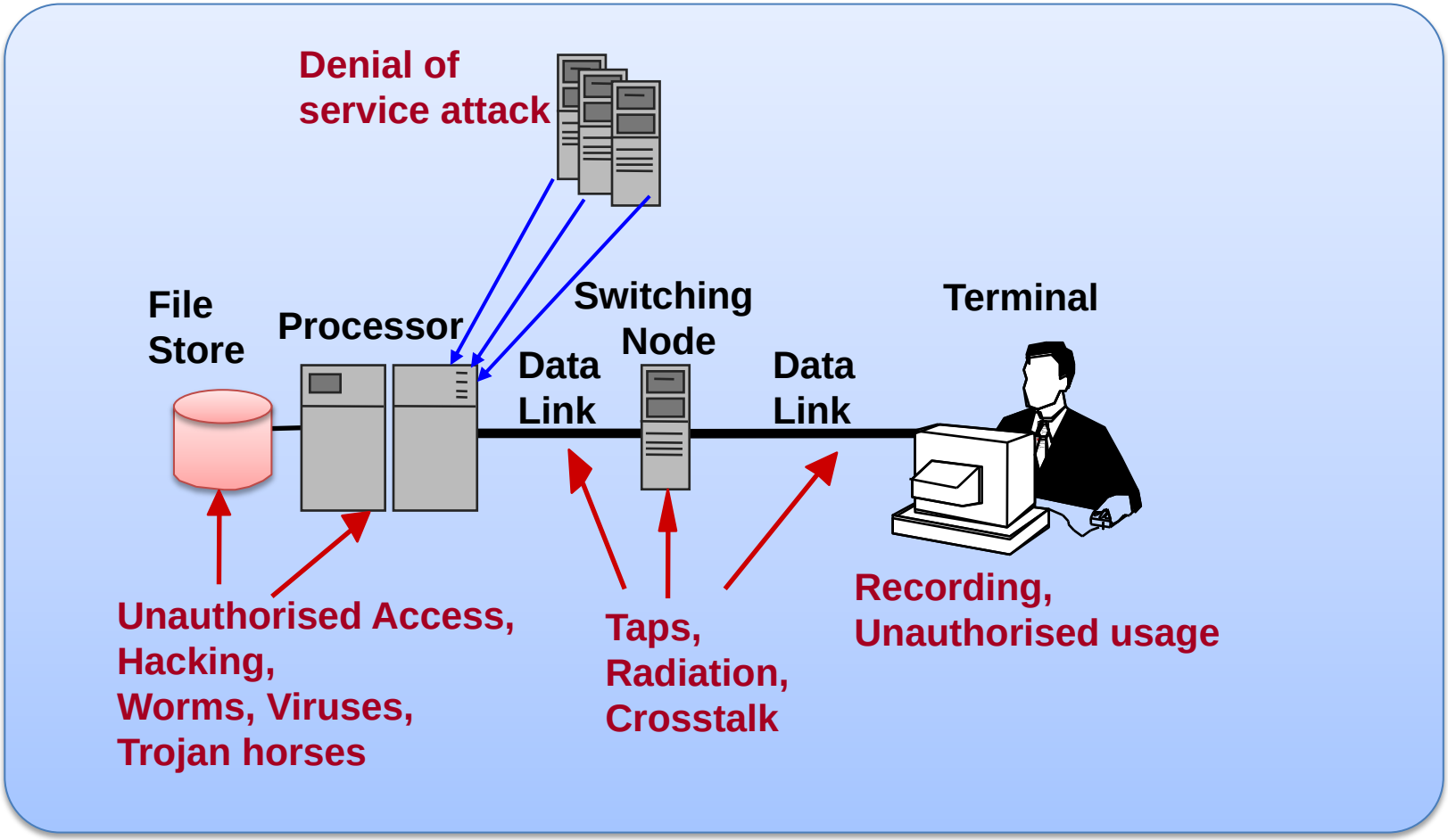
Anandha Gopalan

a.gopalan@imperial.ac.uk
http://www.doc.ic.ac.uk/~axgopala

Overview

- Threats
- Access Control
- Cryptography
- Symmetric Key Distribution and Authentication
- Certificates

Threats


Denial of service attack
File Store
Processor
Switching Node
Terminal
Data Link
Data Link
Unauthorised Access, Hacking, Worms, Viruses, Trojan horses
Taps, Radiation, Crosstalk
Recording, Unauthorised usage

Threats

- Information or Resources
  - Theft / copying, disclosure
  - Modification, corruption or fabrication
  - Destruction
- Services
  - Unauthorised utilisation of resources
  - Disruption of service
  - Denial of access to authorised users
- Users
  - Abuse of privilege by legitimate user
  - Masquerading – impersonation of the identity of another authorised user

Large scale networks (and hence distributed systems) cannot be made physically secure

Types of Attacks

- Passive Attack
  - Observe information in network without interference
  - Message content – break confidentiality
  - Message traffic analysis – frequency, length, source, destination
    - Could have military significance

- Active Attack
  - Modify message contents or message stream
  - Delete, delay, reorder, replay, insert valid or invalid messages
  - Masquerade as authorised user
  - Denial of service by flooding servers with valid requests
  - Passwords gained through passive attack can be used for active attack

Other Security Threats

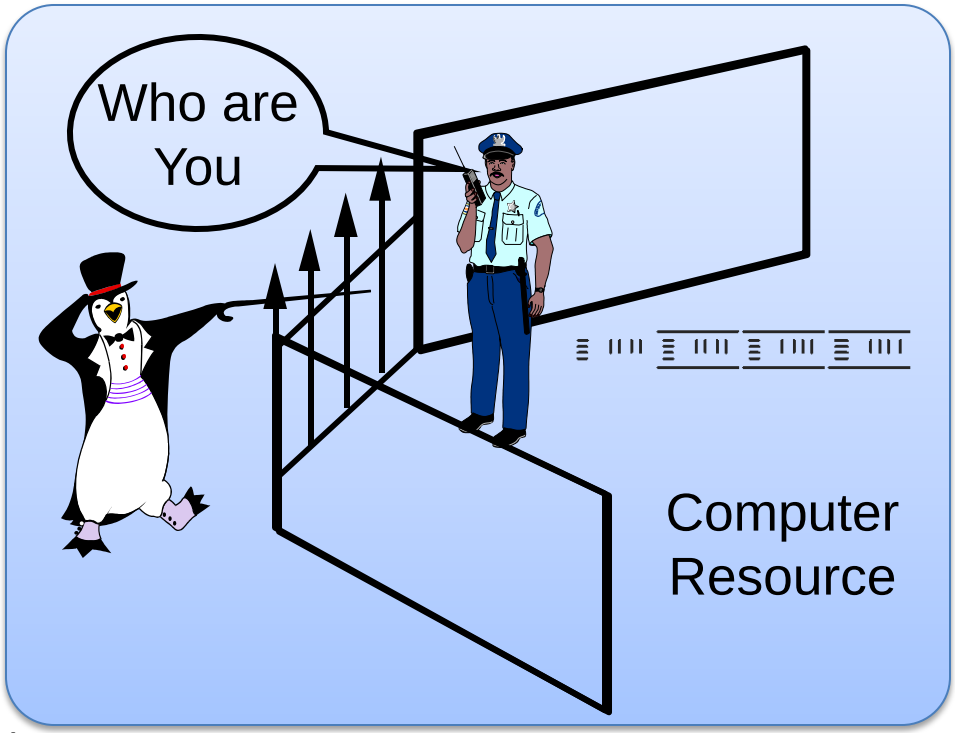
- Unattended Terminal
  - Passer-by can gain access to resources accessible by user
  - Solution – forced log off after timeout
- Man in the Middle attack
  - Relays all interactions between client and server pretending to be the client to the server, and the server to the client
  - Substitutes his own encryption keys for that of the server
- Malware (viruses, trojan horses, worms)
- Compromise Detection
  - Intrusion Detection
  - Anomaly Detection

Security Goals

- Confidentiality
  - Prevent disclosure of information to unauthorised users + prevent analysis of traffic characteristics
- Integrity
  - Prevent modification of information by unauthorised users – includes no duplication, replays, insertions or reordering
- Availability
  - Prevent denial of service e.g. by disruption
- Require:
  - Identification → establishing the identity of the subject
  - Authentication → validity of the identity of sender or server
  - Access Control → control over who has access to services or resources within the system

Identification

- Establish Identity of Subject
- Identification you can use
  - Name or ID provided by user
  - Workstation address
  - Magnetic card
  - Smart card



- Recommendations
  - Use name/initials not numbers
  - Same ID for ALL systems & email
  - Individual accountability → unique ID for each user to identify users in logs and audit trails
  - No sharing of ID

Authentication

- Authentication is Verification of subject Identity
  - Personal identification number (PIN) for magnetic or smart cards
  - Passwords
  - Biometrics e.g. retina or fingerprint scanning
  - Maximum number of authentication attempts (e.g. 3)
  - Logging and investigation of all authentication failures

- Challenge-Response
  - Authentication is often based on challenge-response protocols where the authenticated party needs to prove knowledge/possession of a secret

What is required

- Vulnerability Analysis: identify potential weak elements within system – What is critical to the organisation?
- Threat Assessment: likelihood of a threat occurring which exploits the vulnerability detected
- Risk Analysis: analyse the potential consequences of problems arising from security breach + estimate cost of a successful attack e.g. loss of revenue
- Prevention techniques: what can be done to prevent security breaches and what are their cost?
- Cost benefit analysis: do the consequences of security breaches justify the cost of protection?
- ➢ If security controls cause too much inconvenience or loss in performance they will be bypassed
- Recovery: may be less costly than prevention??

Tools

- Access control
- Cryptography
- Key Management and PKI

Access Control

Access Control

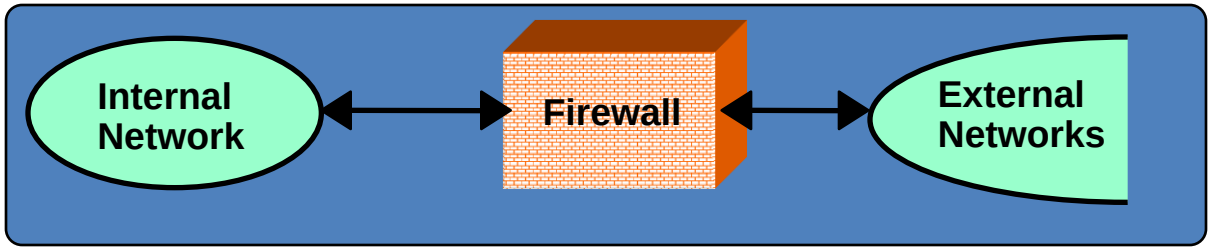
- Ensure that the operations which users or processes can perform on computer resources are done in a controlled and authorised manner

- Needed for
  - Confidentiality → protect resources from being read
  - Integrity → protect resources from being modified
  - Availability → prevent denial of authorised access

- Network level
  - Restrict network traffic flows
- Application level
  - Restrict access to objects and services

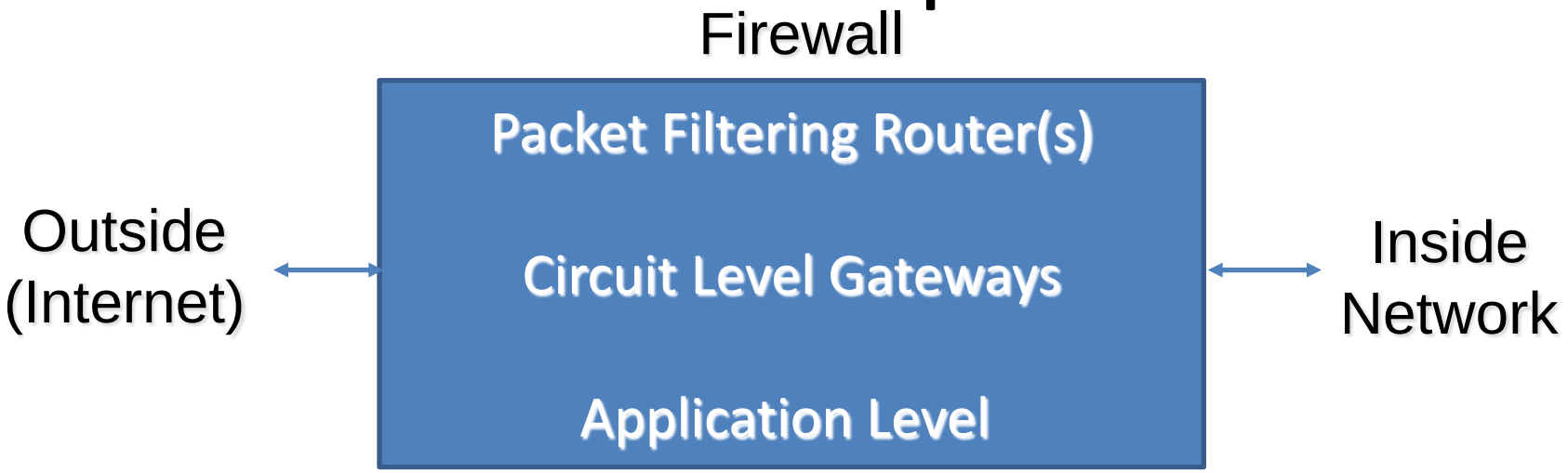
Firewalls

A security gateway between internal and external networks



- Based on packet filters – user defined filtering rules for both incoming and outgoing messages
- Filtering criteria
  - Address – only permit access from selected sites or hosts e.g. remote sites of the same organisation, or collaborators
    - Could be based on combinations of IP and port address
  - Message type – permit incoming Email & HTML (Web) messages but prevent Telnet or FTP

Firewall Components



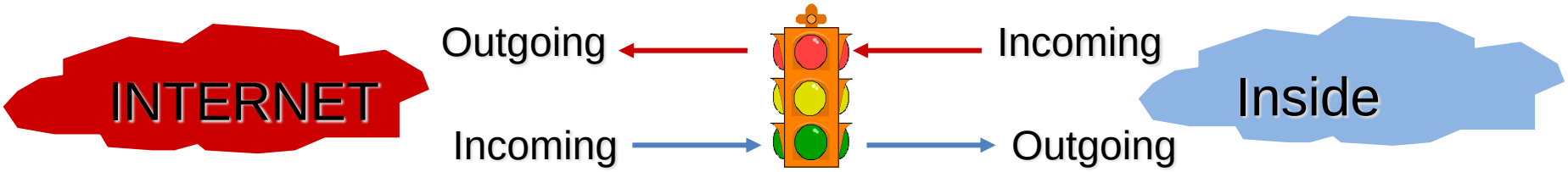
- Packet Filtering Routers (a.k.a. Chokes, Screening Filters)
  - Restrict freeflow of packets between networks
- Gateways
  - Applications (proxies) that provide higher-level processing e.g. authenticating users, "cleaning data", redirecting data, logging, accounting
    - Gateway apps normally run on so-called BASTION HOSTS
- End-to-End encryption between Firewalls is used to create VPNs

Packet Filtering

- Drop Packets based on Source address and/or Destination addresses and/or ports and/or packet header field values
  - Packet Filtering Rules

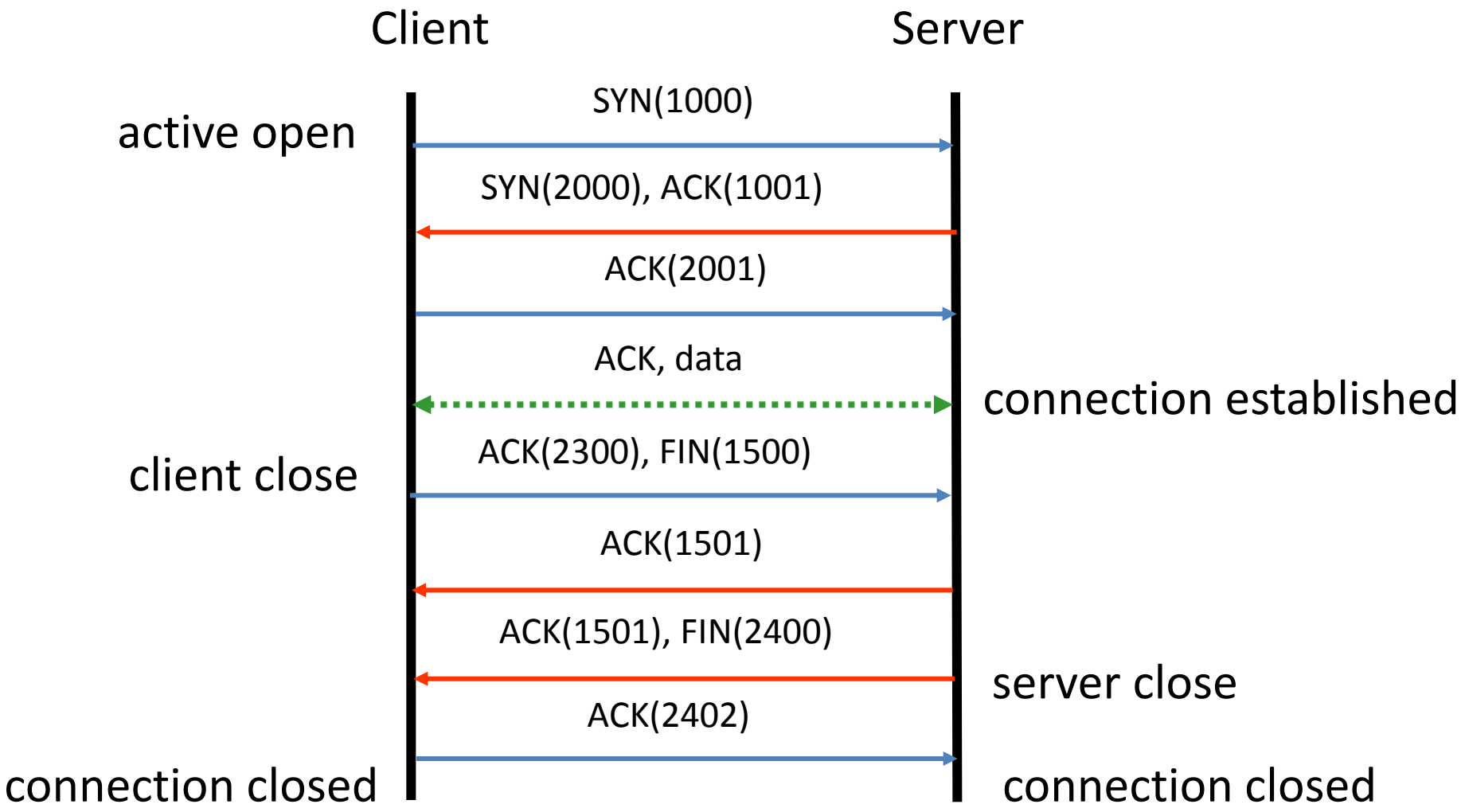
Rule	Dir.	Action	Inside Addr	Inside Port	Outside Addr	Outside Port	Description
1	In	Block	*	*	9.9.9.0	*	Don't let these people in!
2	In	Allow	*	*	6.6.6.6	*	We trust this host
3	*	Allow	1.1.1.7	300	5.5.5.5	300	Very specific access
4	Out	Allow	1.1.1.1	*	*	*	Allow this inside host access
5	Out	Allow	1.1.1.0	*	4.4.4.3	80	Allow access to this service
6	Out	Block	*	*	*	*	Block anything else

Packet Filter Placement



Packet filtering can be performed on incoming or outgoing packets at any interface of the router

TCP Session


Client
Server
active open
SYN(1000)
SYN(2000), ACK(1001)
ACK(2001)
ACK, data
connection established
client close
ACK(2300), FIN(1500)
ACK(1501)
ACK(1501), FIN(2400)
server close
ACK(2402)
connection closed
connection closed

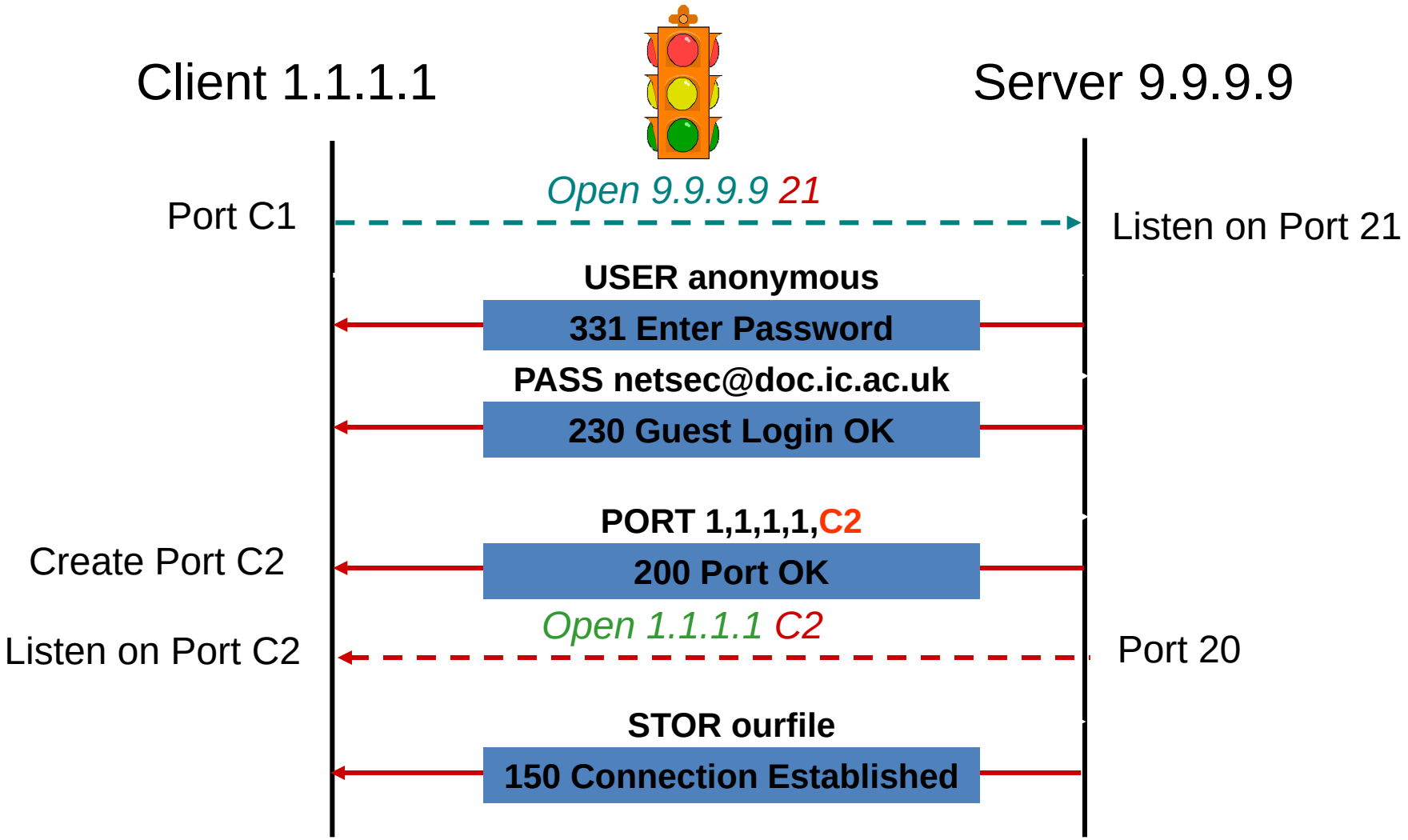
Filtering Rules Revisited

Rule	Dir.	Action	Inside Addr	Inside Port	Outside Addr	Outside Port	Description
1	Out	Allow	*	*	*	25	Mail anywhere (SMTP = 25)

- In TCP, an initial open request packet does not have the ACK bit set, subsequent packets do

Rule	Dir.	Action	Src Addr	Src Port	Dest Addr	Dest Port	TCP Flags	Description
1	Out	Allow	*	*	*	25		Mail out
2	In	Allow	*	25	*	*	ACK	Only replies allowed

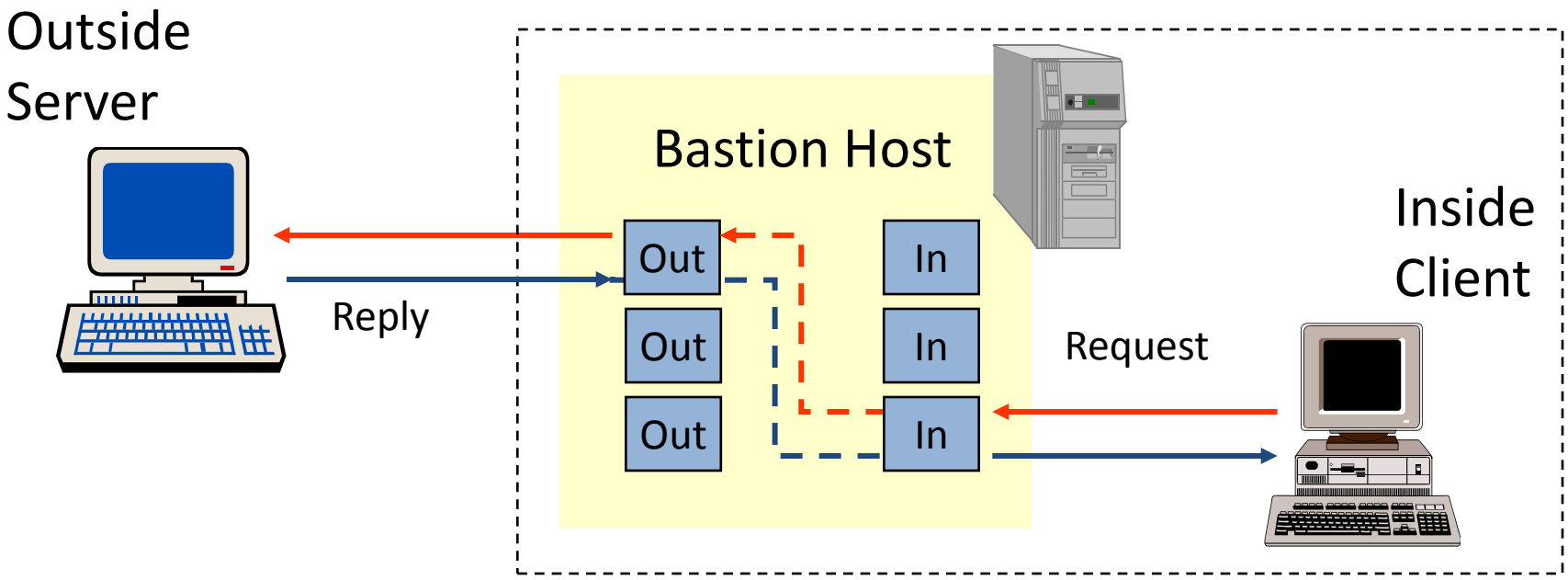
Filtering FTP (Dynamic Callbacks)



FTP Rules

Rule	Dir.	Action	Src Addr	Src Port	Dest Addr	Dest Port	TCP Flags	Description
1	Out	Allow	1.1.1.1	*	9.9.9.9	21		FTP outgoing
2	In	Allow	9.9.9.9	21	1.1.1.1	*	ACK	Only replies allowed
3	In	Allow	9.9.9.9	20	1.1.1.1	>1023		
4	Out	Allow	1.1.1.1	*	9.9.9.9	20	ACK	

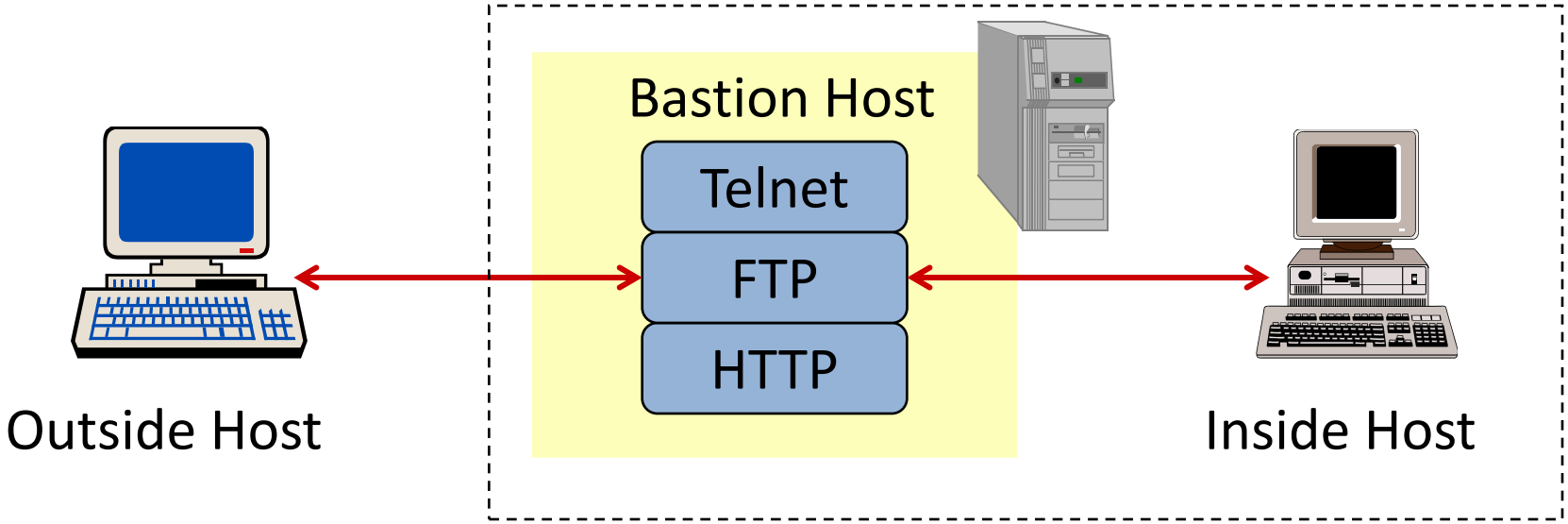
Circuit-Level Gateways



- Relay's connections and maintains connection state. Some can also authenticate users
- Can drop connections based on destination, incorrect connection packets, time, volume, etc.

- Normally used for Inside-to-Outside connections. Client normally recompiled to connect to CL-Gateway port
- Good for auditing / accounting

Application-Level Gateways



- Like CL-GW's but application-specific (app-knowledgeable)
- Can block/filter/report based on app-level msg. content. Can scan for data leaks, viruses, etc. Can rewrite data

- Can be configured to limit application features
- More processing overhead than CL-GW's

Bastion Host

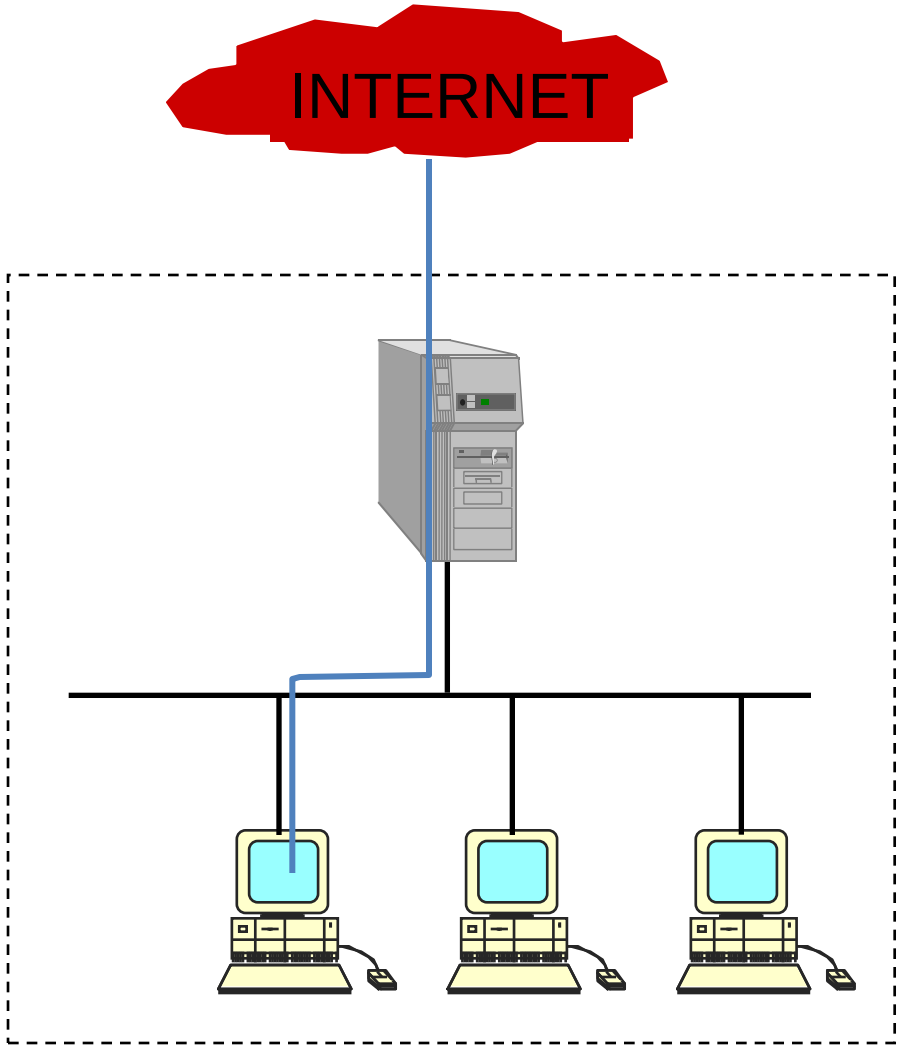
- Runs application-level and circuit-level gateways
- Can run other servers too
- Performs auditing / accounting
- Should run a "Trusted"/Secure OS
- Administer via a dedicated terminal

Minimal OS

- Remove inessential applications, utilities, services, e.g. cc, awk, sed, ld, X11
- Set file permissions, turn on file quotas, process limits etc.
- No regular user accounts
- No NFS mounts
- Make filesystems read-only if possible

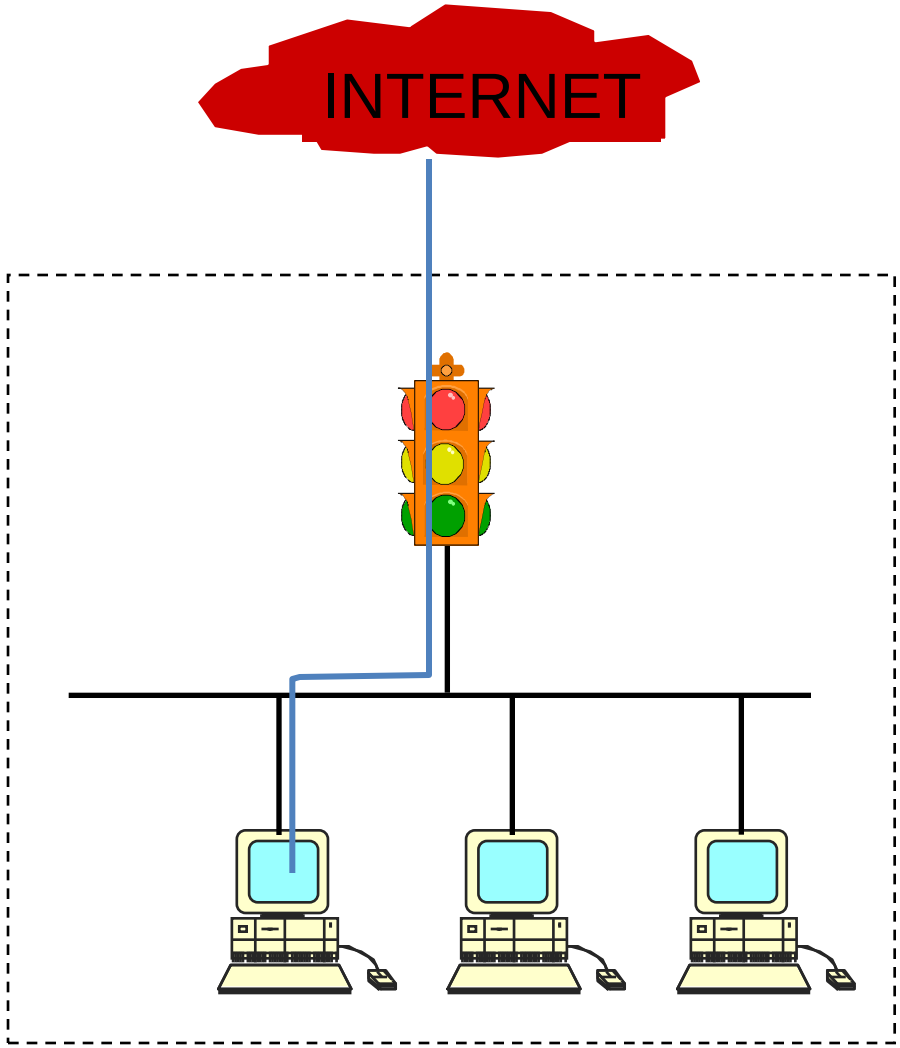
Firewall with One Bastion Host

- Earliest example of Firewall
- Bastion host acts as filter and gateway. Also runs proxies for permitted services
- Note: Bastion Host must not automatically forward packets
- On some Bastion Host Firewalls, users were allowed to login directly onto bastion host (bad!)



Firewall with one Packet Filter

- Block all packets for “services” that are not used
- Block all packets that explicitly set IP source routing or other unusual options
- Block access to/from particular sites, networks
- Allow incoming connections to predetermined services, block others
- Optionally allow internal hosts to initiate outgoing TCP connections



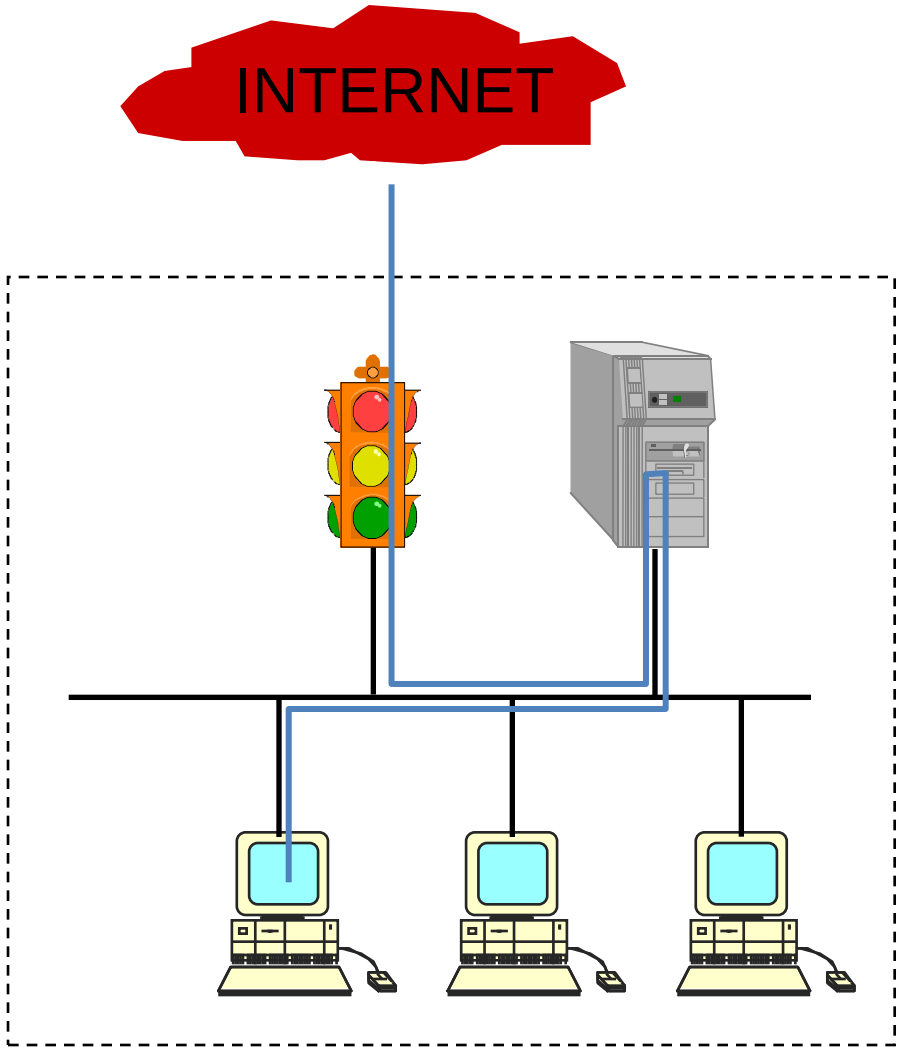
Screened Host Architecture

PACKET FILTER

- Block packets for services that we do not wish to cross firewall, and packets with IP source routing
- Block packets for internal network
- Only pass packets for which the Source or Destination IP address is the Bastion Host

BASTION HOST

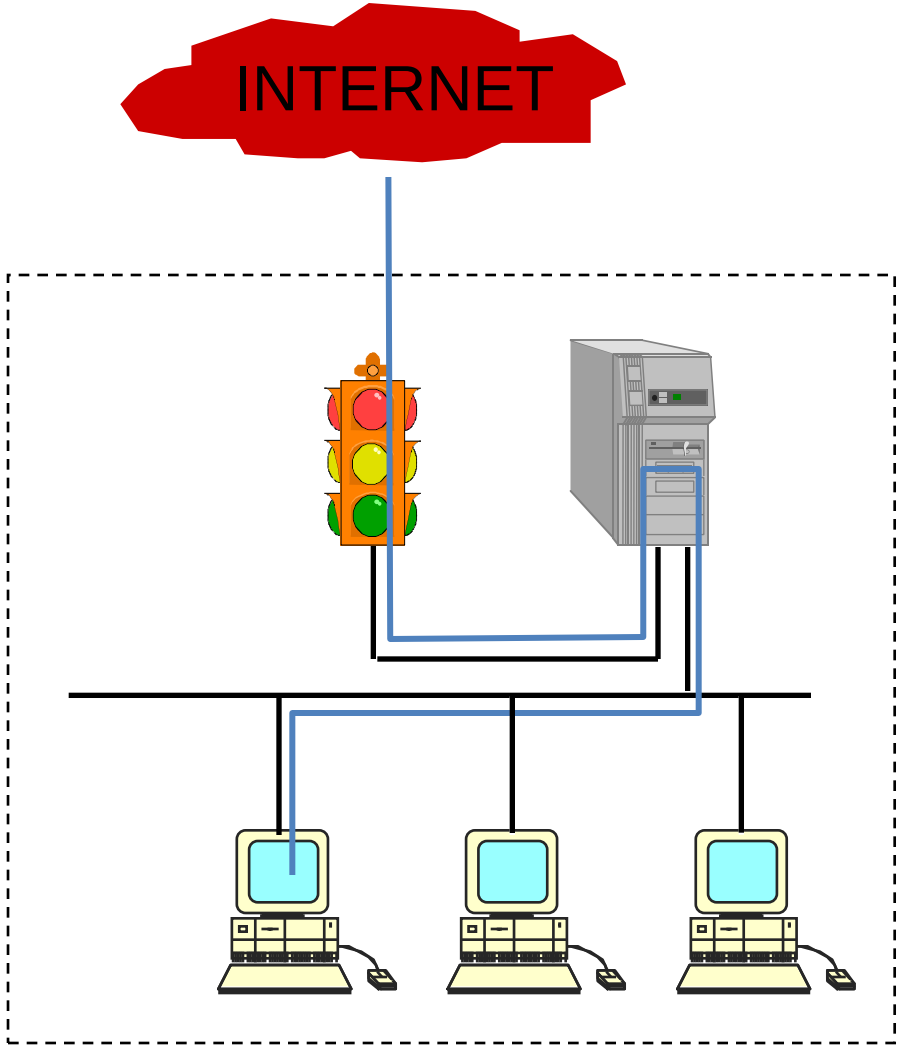
- Run application-level gateways, e.g. web server, mail server
- Run circuit-level gateways to allow users on the inside to access servers on the internet



Screened Host (Dual-Homed) Architecture

DUAL-HOMED BASTION HOST

- What if the packet filter is compromised?
- For greater security we can use a dual-homed bastion-host, i.e. a host that has 2 network-interfaces
- Now any intruder who gets past the packet filter will also need to get past the bastion host

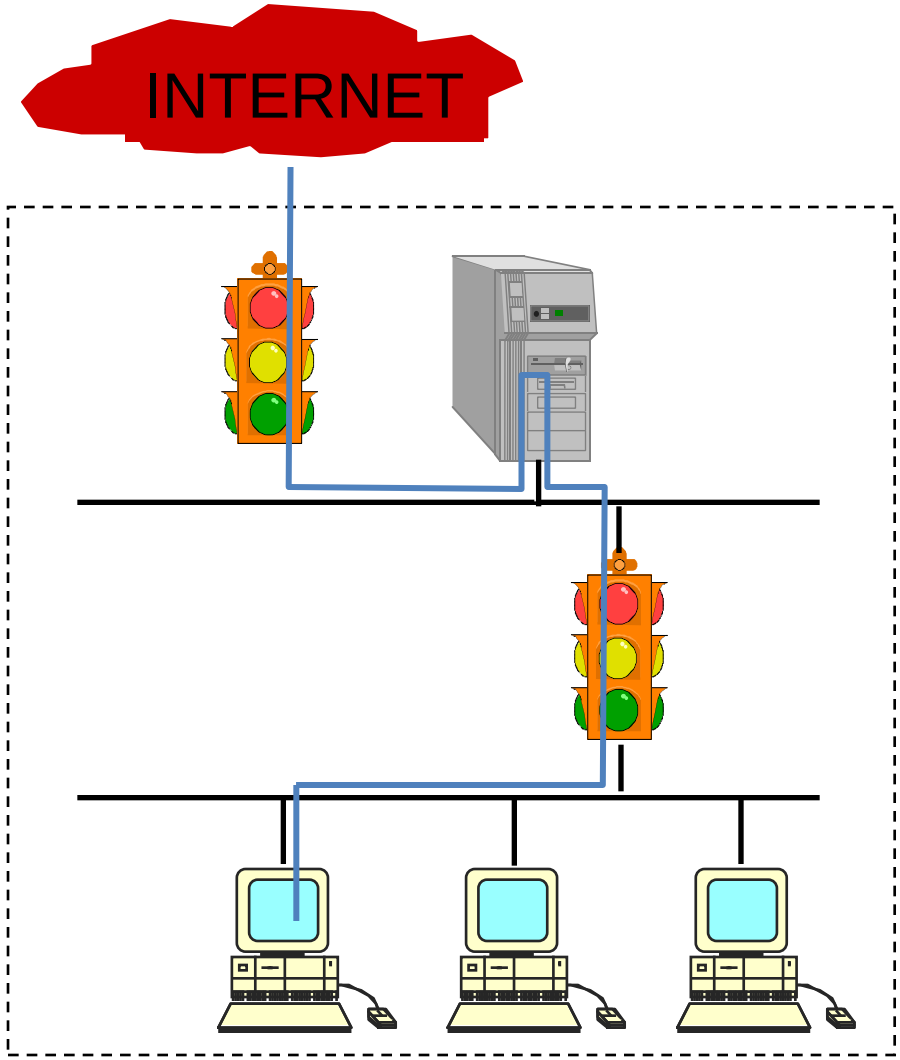


Screened Subnet Architecture

2nd inner packet filter prevents attack from compromised bastion host

INNER PACKET FILTER

- Block packets for services that we do not wish to cross firewall, and packets with IP source routing
- Block packets addressed for outer filter
- Only pass packets for which the Source and Destination IP address is the Bastion Host and for which the ports are for defined proxies on the gateway
- Block everything else



Access Control Policy Types

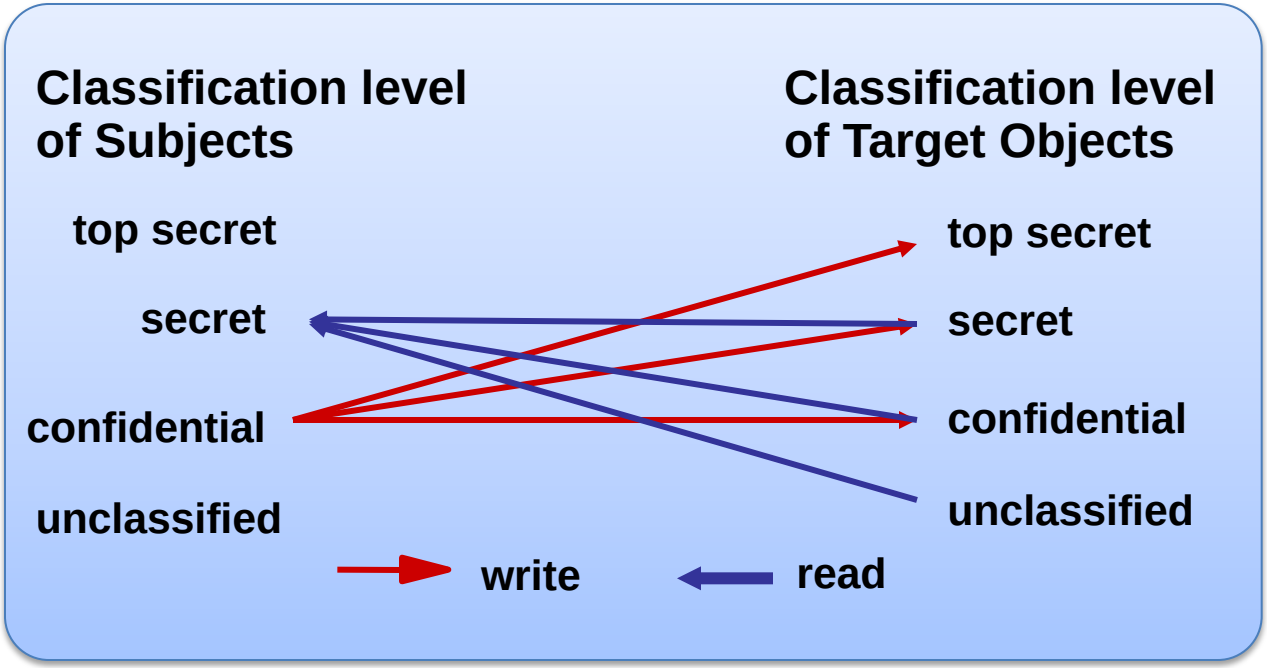
- Mandatory Policy
  - The security rules which are built into the system and cannot be changed by users i.e. they cannot be trusted to enforce security policy
  - E.g. no information from a high security level can flow to a low security level
    - Subjects labelled with clearances defining their privileges
    - Targets labelled with classification defining sensitivity
- Discretionary Policy
  - The security rules which a user or manager may choose to apply e.g. who can access your personal files
  - Typically defined by identity based rules where access decided on identity of subject

Military Security Policy

- Based on Security level classifications (labels)
- Mandatory labelling of all documents with classification level
- Regulate control of classified information and prevent access by unauthorised people → emphasis on confidentiality & controlling flow of information
- Mandatory controls constrain user so that any action taken conforms to security policy – very little flexibility
- Discretionary aspect: user can further restrict access, but cannot re-label to a lower classification

Bell-Lapadula Security Model

- Typical military security levels: unclassified, confidential, secret, top secret
- Categories of information e.g. nuclear, naval, planes
- Subject Label (clearance) = (Level {categories})
  e.g. secret {nuclear, planes}
- Target Label = (level {1 category})



Write up
Read down

Bell-Lapadula Security Model

- Simple Security condition
  - Subject s may only read from target t,
    if (s.level $\geq$ t.level) & (t.category $\in$ {s. categories})
    - i.e. read information of permitted category at same or lower classification e.g. officer cleared for secret cannot read top secret, but can read all lower classified documents

- ★ Property
  - Subject s can write to a target t
    if (s.level $\leq$ t.level of t) & (t.category $\in$ {s. categories})
    - i.e. write to documents at higher classification → prevents copying information from higher to lower levels
    - Can only have read/write access to documents at same level
- Anomaly ?

Authorisation

- Specification of security policy in terms of what resources a subject can access and what operations can be performed upon them
  - Authorisation should be in terms of user roles rather than user identities i.e. Manager of Personnel rather than Joe Bloggs → simpler when people move jobs
  - Default to no access i.e. require explicit authorisation to permit access

- **Principle of least privilege**
  - Any subject or user object should be given the minimum access rights required to permit it to carry out its assigned task. This principle is violated if a server has to take on the user's identity in order to perform a function on its behalf

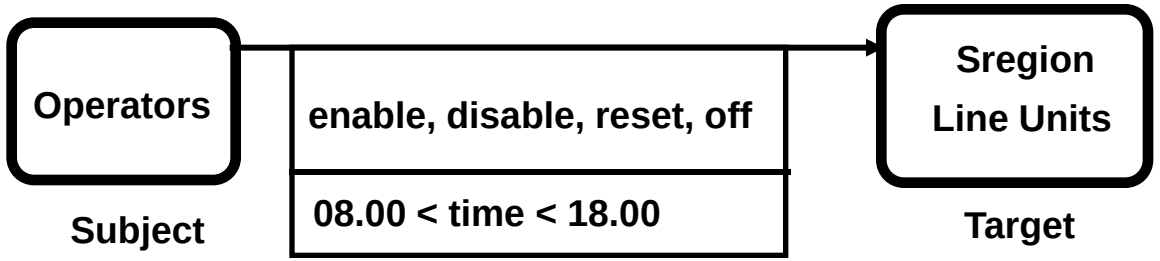
Authorisation Policy

- Defines what activities a subject is permitted or not permitted (prohibited) to do to a target

- Specify:
  - Modality – permit or forbid
  - Subject Domain e.g. section, department or manager
  - Target Domain e.g. directory
  - Set of permitted operations
  - Constraints – time of day, day of week, or a predicate on the value of a subject or target attribute, e.g. location of subject

Authorisation Policy

Examples




```
inst auth+  { subject Operators;  target <line_unit> Sregion;
     action enable, disable, reset, off;
     when time.between ( "08.00", "18.00") ; }
inst auth- {subject  students; action  reboot;
     target  teaching_workstations ; }
inst auth+ { subject  s = managers; action read, write;
     target development_directory ;
     when s.location = planning office;  }
```

Implementation

- Access Control List (ACL) attached to target domain
  cf. column of Access Matrix
- Set of Access Tickets or capabilities held by user domain
  cf. row of Access Matrix

Access Control Implementation: Access Matrix

Object / User/ Subject	F1	F2	Printer	O1	O2
mss	read, write		print	a, b c	i, j
jnd	read	execute	print	b, c	k
jk		read	print	c	i, j, k
xyz	append	read, write		a, b	i

Access Control Lists

- Let S = set of subjects, O = set of objects, R = set of rights. The Access Control List for Object o is a set of pairs: { (s, r) : s ∈ S, r ⊆ R} such that a subject s can use any associated right in r to access object o

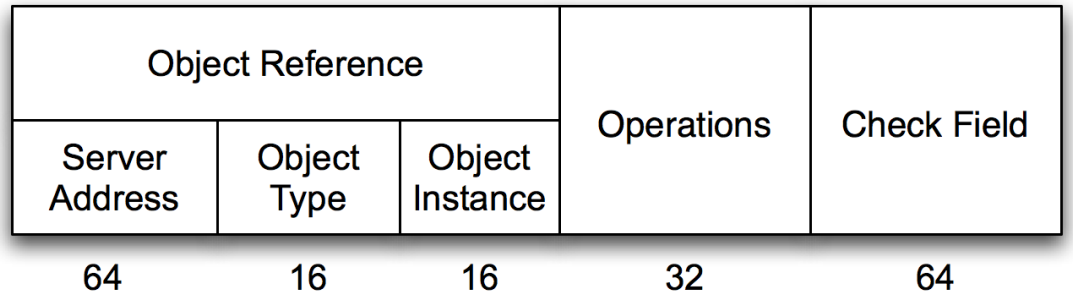
- Usually assumes a default negative policy i.e., no entry in the ACL = no right to access

- Subjects can be identified as users, groups, roles, etc. Some implementations also define a wildcard '*'

- Conceptually corresponds to a column in the access matrix

Capabilities

- Conceptually a capability corresponds to a row of the access matrix
- Let S = set of subjects, O = set of objects, R = set of rights. A capability list for subject S is a set of pairs: {(o, r) : o ∈ O, r ⊆ R} such that s can use any associated right in r to access o
- In practice a capability is a token or ticket given to or otherwise associated with the subject. Subject must not be able to alter capability
- Usually implemented through protected OS objects or cryptography

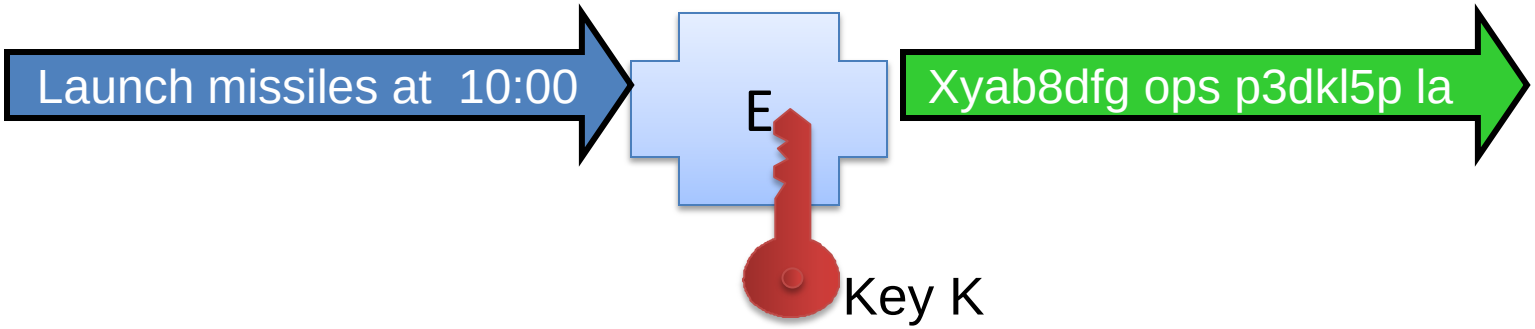


Capabilities

- Check field is an encrypted version of the permitted operations field and a suitable random constant. It prevents forging the capability
- Checking access rights is simpler; burden of choosing permission is put on the client
- Creating and copying capabilities must be done through protected operations of the operating system (copying capabilities implies the ability to give rights)
- Removing all access rights of a subject is simpler. Removing a right from all subjects is difficult
- Revocation of an access right is more difficult. Use expiry time to mitigate the effect

Cryptography

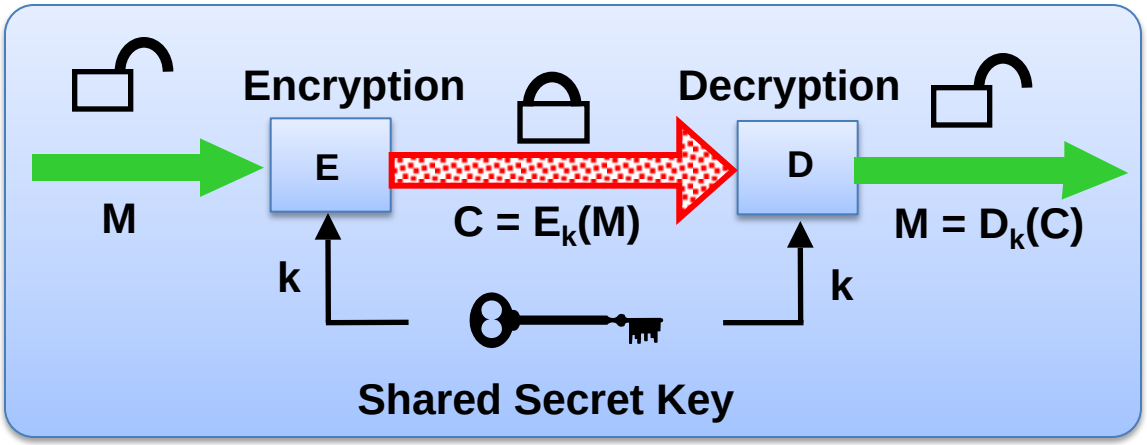
Cryptography



- **Encryption** is a transformation of information based on:
  - Substitution e.g. table look up
  - Transposition e.g. exchange bytes 1 & 3, 2 & 4 etc.
  - Combine with a key which specifies what substitution or transposition to use → Encryption *Function E + Key k*
- **Decryption**
  - Inverse of encryption to obtain original information
  - Computation time required to decrypt without the key makes it impractical but not impossible

Secret Key (Symmetric) Cryptography

Single Secret Key

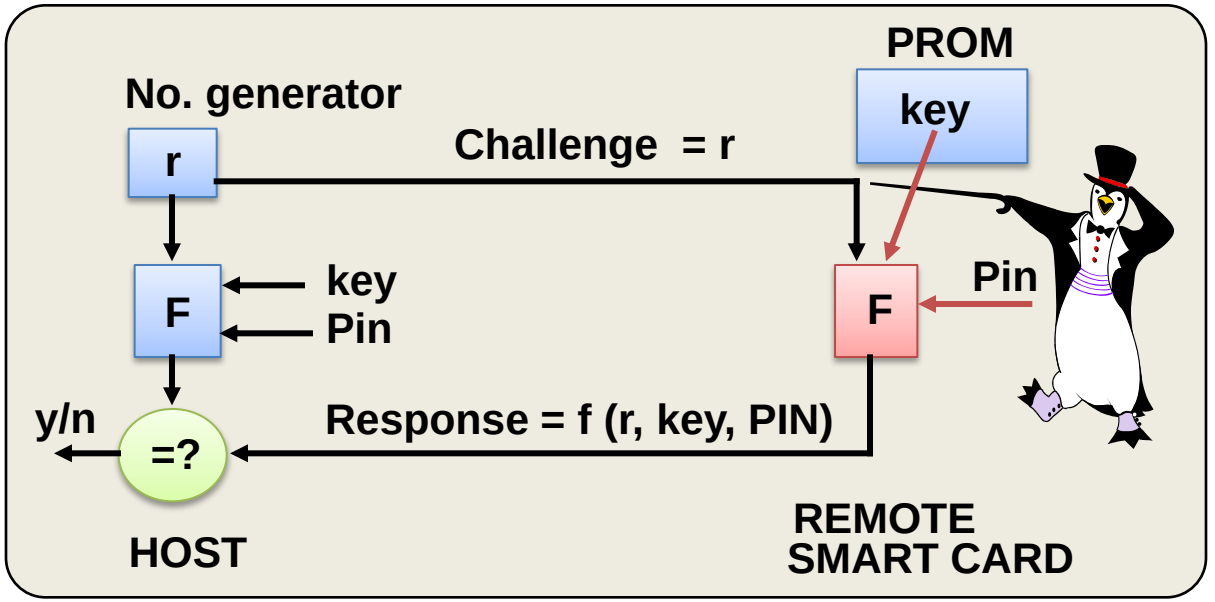


- Basis for Data Encryption Standard (DES)
  - Same algorithm applied for Encryption and Decryption i.e. E = D
  - 56 bit key applied to blocks of 64 bits of data – easily cracked
- Advanced Encryption Standard (AES) selected in March 2001 after public competition
  - Rijndael Cipher from Belgium → 128, 192 or 256 bit keys
- Problems of key management

Remote Access Authentication

Smart Cards

- Embedded microprocessor + storage
- Preprogrammed with Key in PROM
- Require possession + knowledge of PIN



Challenge / Response Protocol

- Host generates random number r which is sent to Smart Card. Card uses one way function F, seeded with cryptographic key K and PIN number to calculate response which is sent back to host. Host uses same function to check response
- Not susceptible to replay

Key Management

- Secret key required for each partner or even session → Key distribution problem
- Hierarchical Key Usage
  - Distribute master key by courier e.g. monthly
  - Use master key to encrypt new keys sent out e.g. daily
  - Use current day key to encrypt session key
- Use session key distribution server e.g. Kerberos

Public Key Encryption

- Rivest, Shamir, Adleman (RSA) Algorithm

- Use two keys
  - Public Key (K_{pub}) sent out over network and stored with name servers – used by sender for encryption
  - Secret Key (K_{sec}) – used by recipient for decryption
  - Cannot deduce secret key from public key

- Property → $D_{K_{priv}}(E_{K_{pub}}(M)) = M$

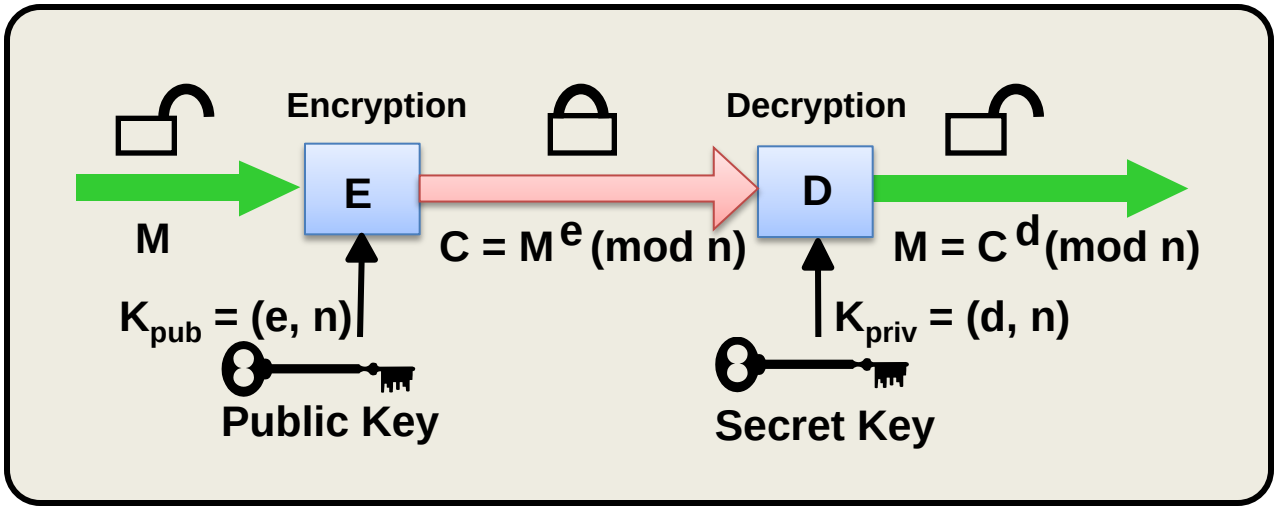
  - i.e. decryption of encrypted message yields the original message – symmetric encryption & decryption

RSA Algorithm

- Choose two large primes at random (p, q)
  - For our example, we pick 'small' p = 3, q = 11
- Calculate n = p * q ➔ n = 33
- Calculate ϕ (n) = (p – 1) * (q – 1) ➔ ϕ (n) = 2 * 10 = 20
- Select at random the encryption key e, such that, 1 < e < ϕ (n) and gcd (n, ϕ (n)) = 1 ➔ e = 7
- Select at random the decryption key d, such that, e * d mod ϕ (n) = 1 and 1 < d < ϕ (n) ➔ d = 3

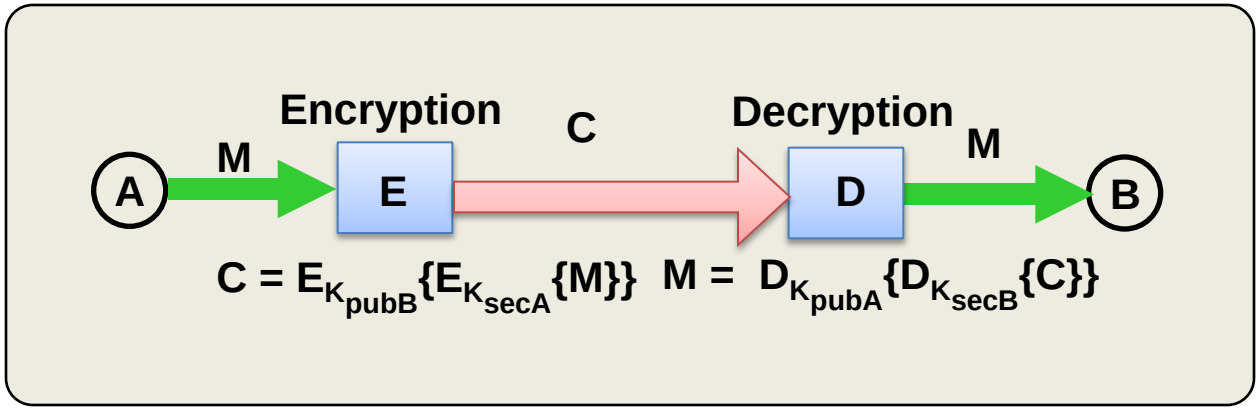
- Public key is thus (e, n)
- Private key is thus (d, n)

Public Key Encryption



- Simple Key management → no secret keys to distribute
- Computation intensive → poor performance
- Use public key system for distribution of secret session keys, and session keys for encrypting messages

Public Key Signatures



- Use public key encryption where: $D_{K_{pub}}(E_{K_{priv}}(M)) = M$ and $D_{K_{priv}}(E_{K_{pub}}(M)) = M$
  - A uses a secret key to encrypt a message
  - B receives the message and decrypts it with A's public key
  - If the decryption is successful it must have been encrypted by A, i.e. "signed" by
    - Authenticates sender
- Can be decrypted by anyone with A's public key
  - Encrypt with B's public key – only B can decrypt

Public Key Signatures

- No verification of time or prevention of replay
  - X could capture message and repeat it e.g. if message was from point of sale terminal in Company X to A's Bank to transfer £1000 to X's account
  - A can pretend secret key K_{secA} was stolen and deny sending message
- Need to store complete encrypted message for later verification
- *What is needed?*

Message Digest

- A one way function on a message → a code which can be used to check its integrity e.g. checksum or hash function
- Detects message modification but does not provide secrecy
- Sender computes digest, appends to message and receiver re-computes to check
  - Given message m it is easy to compute H(m)
  - Given H(m) it is impossible to compute m
  - No 2 messages can generate same H(m)

Message Digest

- Use cryptographic hashing or encrypt digest
- Hashing is faster than encryption e.g. MD5 or Secure Hashing Algorithm (SHA)
- Pad message to multiple of 512 bits & mangle blocks of 512 bits to produce 128 bit (MD5) or 160 bit code (SHA)

Certification Problem

- Alice has generated a new electronic work of art in the form of a large 50MB file. She wants to be able to prove that she is the originator of the work so requires a proof from an origin authority (OA)

- Using public key cryptography and message digests show
  I. What information Alice sends to the OA
  II. The response from the OA

Certification Solution

Alice $\rightarrow$ OA: $\quad$ Alice, $K_{privA}\{Alice, H(file)\}$

Alice sends a digest of the file to obtain the digital signature, encrypted with her secret key. OA uses Alice's public key to decrypt and prove it came from Alice

OA$\rightarrow$ Alice : $\quad$ $K_{secOA}\{Alice, H(file), t\}$

Digital signature includes Alice's ID and time stamp of when it was generated. This can be checked with the OA's public key

Digital Signature

- Needed for legally binding documents. E.g. Joe sends message to stockbroker to buy 10,000 shares in HAL Plc. which goes bust next day

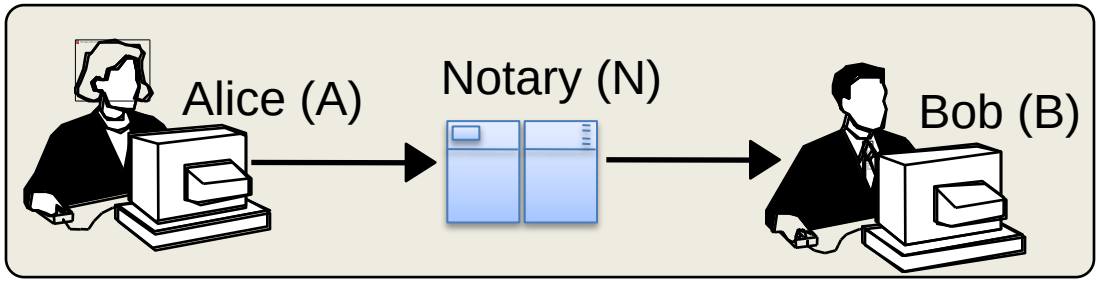
- Joe may deny sending message or stockbroker may change number to 100,000 to offload other shares he had bought for himself
  - Verify author, date and time of signature
    - Non-repudiation of origin
  - Verify contents at time of signature
    - Receiver or someone else cannot modify contents or forge message claiming it came from sender

Digital Signature

- Signature must be verifiable by third party to resolve dispute
- Signature must be unforgeable
- Must be practical to retain copy of digital signature in storage

- Need Notarisation Service (Arbiter) to prevent non-repudiation by sender or subsequent claims of loss of secret keys

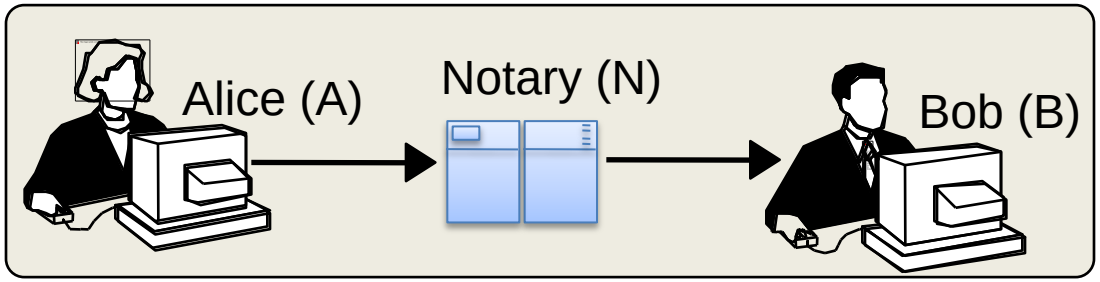
Notarised Signatures



- Assume source name and address is included in every message – why?

- A → N: A, B, $K_{AB}\{m\}$, $K_{AN}\{A, H(K_{AB}\{m\}), T_A\}$
  - Message m is protected by K_{AB} known only to Alice and Bob so cannot be read by Notary. T_A is A's timestamp
  - Notary validates encrypted message by checking hash digest
  - Notary inserts its timestamp T_N in message

Notarised Signatures



- N → B: N, $K_{BN}\{A, T_A, T_N\}$, $K_{AB}\{m\}$, $K_{AN}\{A, T_A, T_N, H(K_{AB}\{m\})\}$
  - Bob knows timestamp of generation and signing so can check for freshness
  - The notary's signature $K_{AN}\{A, T_A, T_N, H(K_{AB}\{m\})\}$ is stored – Bob cannot decrypt it
  - Only m needs to be stored as $K_{AB}\{m\}$ can be recreated

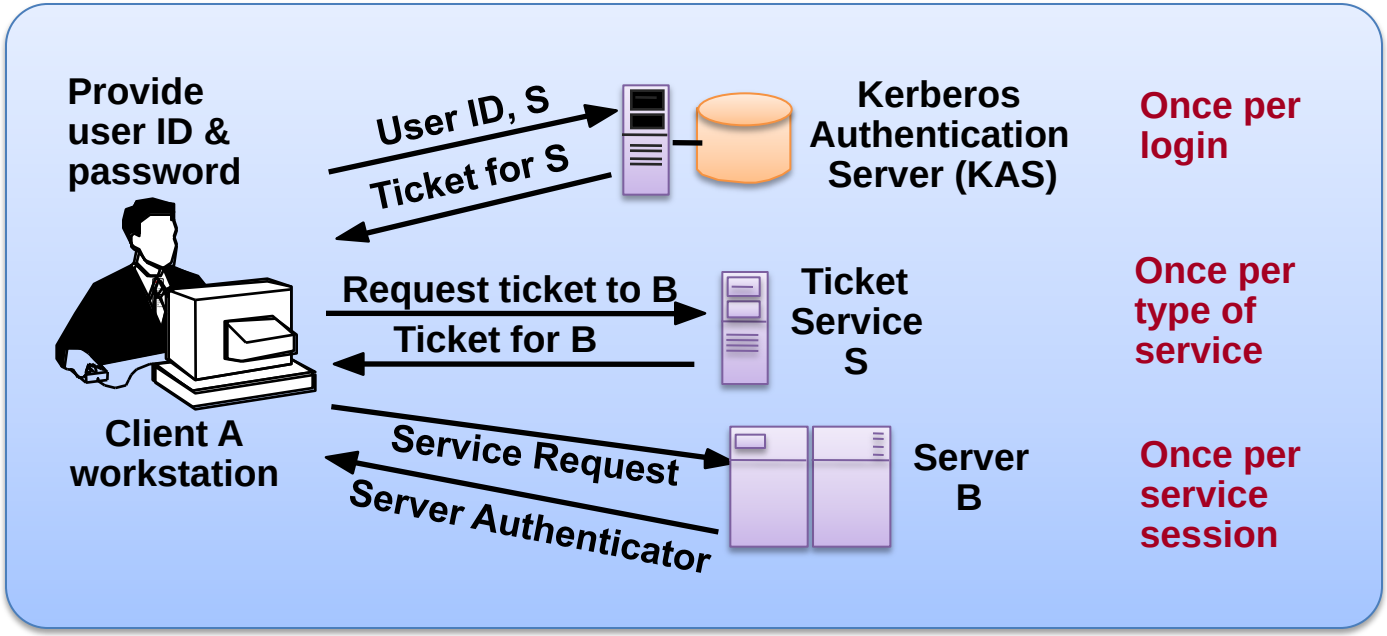
Notarised Signatures

- In case of dispute Bob sends the following to the Notary
  - B → N: B, K_{BN} {A , T_A, K_{AB} {m}}, K_{AN}{ A, T_A, T_N, H (K_{AB}{m})}
  - The notary can then verify the time, the sender and that the digest of the encrypted message is valid so m has not been changed
- Proof of receipt
  - B → N: B, K_{BN} {A, H (K_{AB} {m}), T_B}
  - N holds the original message until the receipt is received and checks the digest H (K_{AB}{m}), adds it's timestamp and B's identity, then forwards the receipt to A
  - N → A: B, K_{BN} {A, B, H (K_{AB}{m}), T_B, T_N}
  - A holds the receipt which it cannot decode or modify, but could be sent to N in case of dispute

Symmetric Key Distribution and Authentication

Kerberos Authentication Service (V4)

- Separate authentication and ticket granting service
- Users and Service providers must register with Kerberos



Kerberos Authentication Service (V4)

- KAS stores identities, server private keys and encrypted user passwords used to generate private user keys
  - Must be secure + master/slave redundancy
- Ticket service can be replicated – does not hold secret data
- Use time stamps to detect replay
  - Lifetime of tickets issued by ticket server must be less than the lifetime of ticket originally issued by Kerberos
  - Servers accept tickets within limited window of timestamp to prevent replay or masquerading
  - Require clock synchronisation (within minutes)

Tickets and Authentication

- Ticket for Server X
  - Secure transfer of authenticated identity of user to server, plus optional authorisation data that can be used for access control

- $T_{AX} = K_x \{A_{name}, A_{realm}, A_{addr}, K_{AX}, TS, TL, X_{name}\}$
  - TS = ticket timestamp
  - TL = ticket lifetime
  - Can be reused
  - Generated by KAS for ticket service, or by ticket service for Server B
  - K_{AX} = session key

Authenticators

- Authenticator for Server X
  - Proves identity of user presenting ticket is same as that to whom ticket was issued

- $X_{AX} = K_{AX}\{A_{name}, A_{realm}, \text{Appl. data, TS}\}$
  - Can only be used once when session with X is established
  - Generated by client A

Kerberos Interactions

- A logs in and provides userid & password
  - Workstation uses password to generate K_A then discards password

- A → KAS: {Options, A_{name}, A_{realm}, S_{name}, TS }
  - KAS generates session key K_{AS} & ticket T_{AS} for ticket server S
  - KAS uses A's password to generate Key K_A

- KAS → A: K_A {K_{AS},TS_2, TL, T_{AS}}
  - Only client A can decrypt this to obtain ticket T_{AS}
  - Use K_{AS} to create authenticator X_{AS} for S

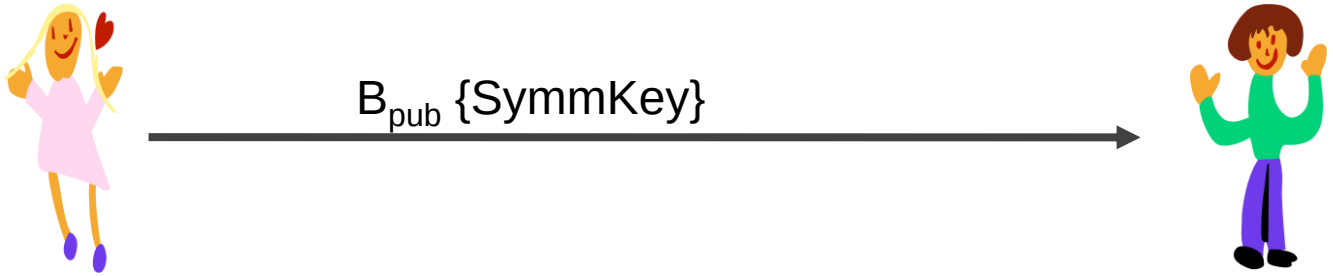
Kerberos Interactions

- A → S: $\{B_{name}, T_{AS}, X_{AS}\}$
  - Decrypt T_{AS} to get session key K_{AS} - check authenticator & ticket
  - Generate session key K_{AB} and ticket T_{AB}
- S → A: $K_{AS}\{ K_{AB}, B_{name}, TS, T_{AB}\}$
  - Decrypt to get ticket and session key for B
  - Create authenticator for B
- A → B: $\{T_{AB}, X_{AB}\}$
  - Decrypt ticket to get session key, check authenticator, generate response by adding 1 to TS in authenticator → this authenticates server B to A
- B → A: $K_{AB}\{TS + 1\}$
  - Only server B can generate this response

Certificates and Public Key Infrastructure

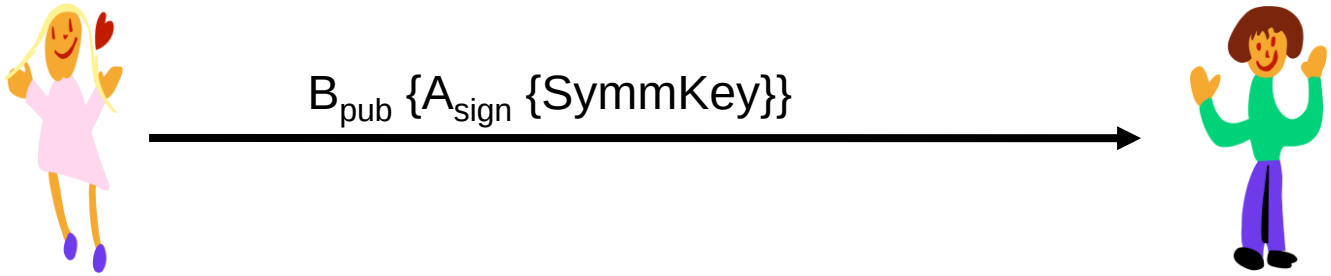
Public Key Dist. of Symmetric Keys

Take 1



- Assumes Alice knows Bob's public key
- Bob has no idea he is speaking with Alice

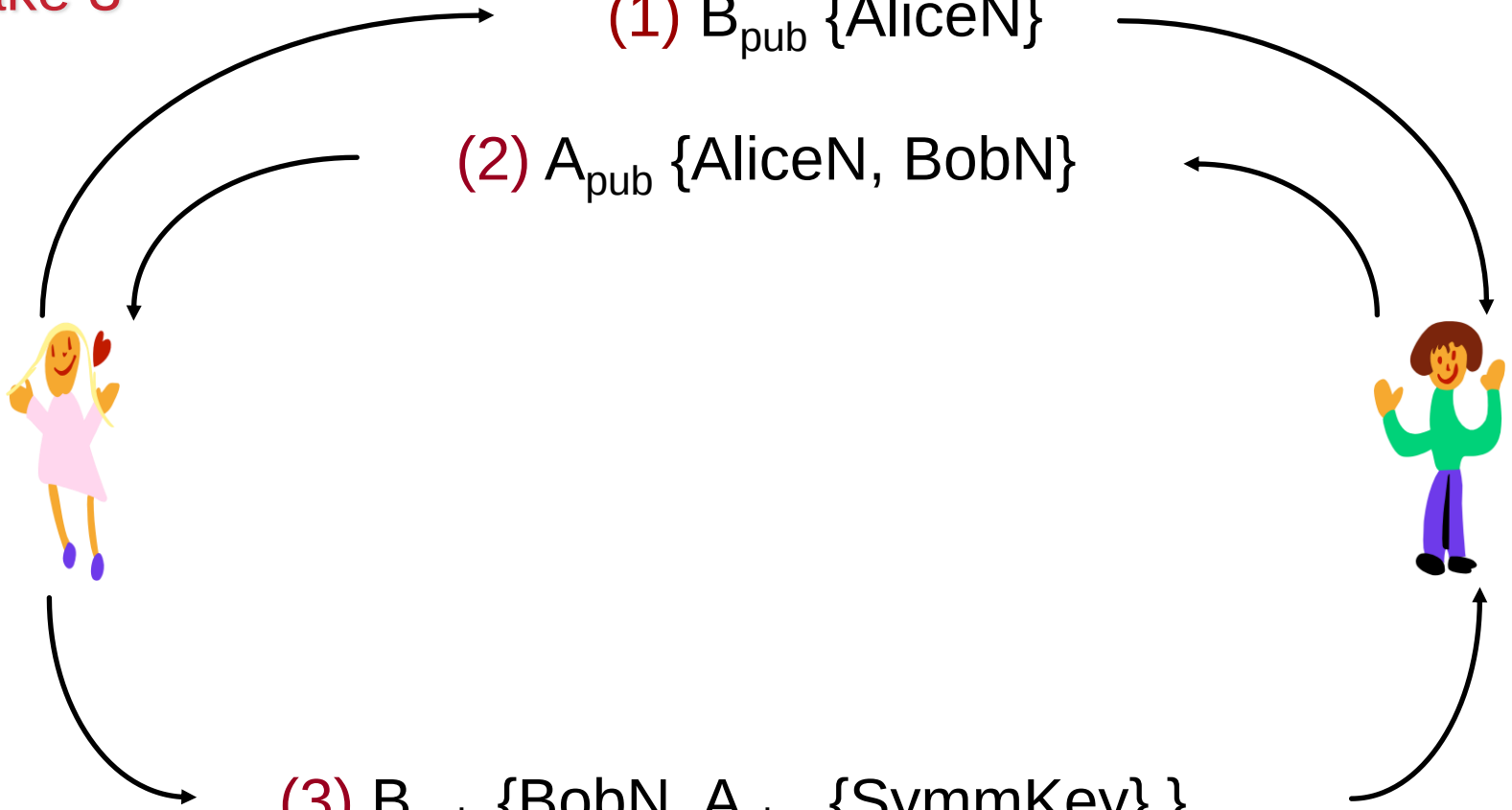
Take 2



- Assumes both Alice and Bob know each other's public key
- No idea of freshness

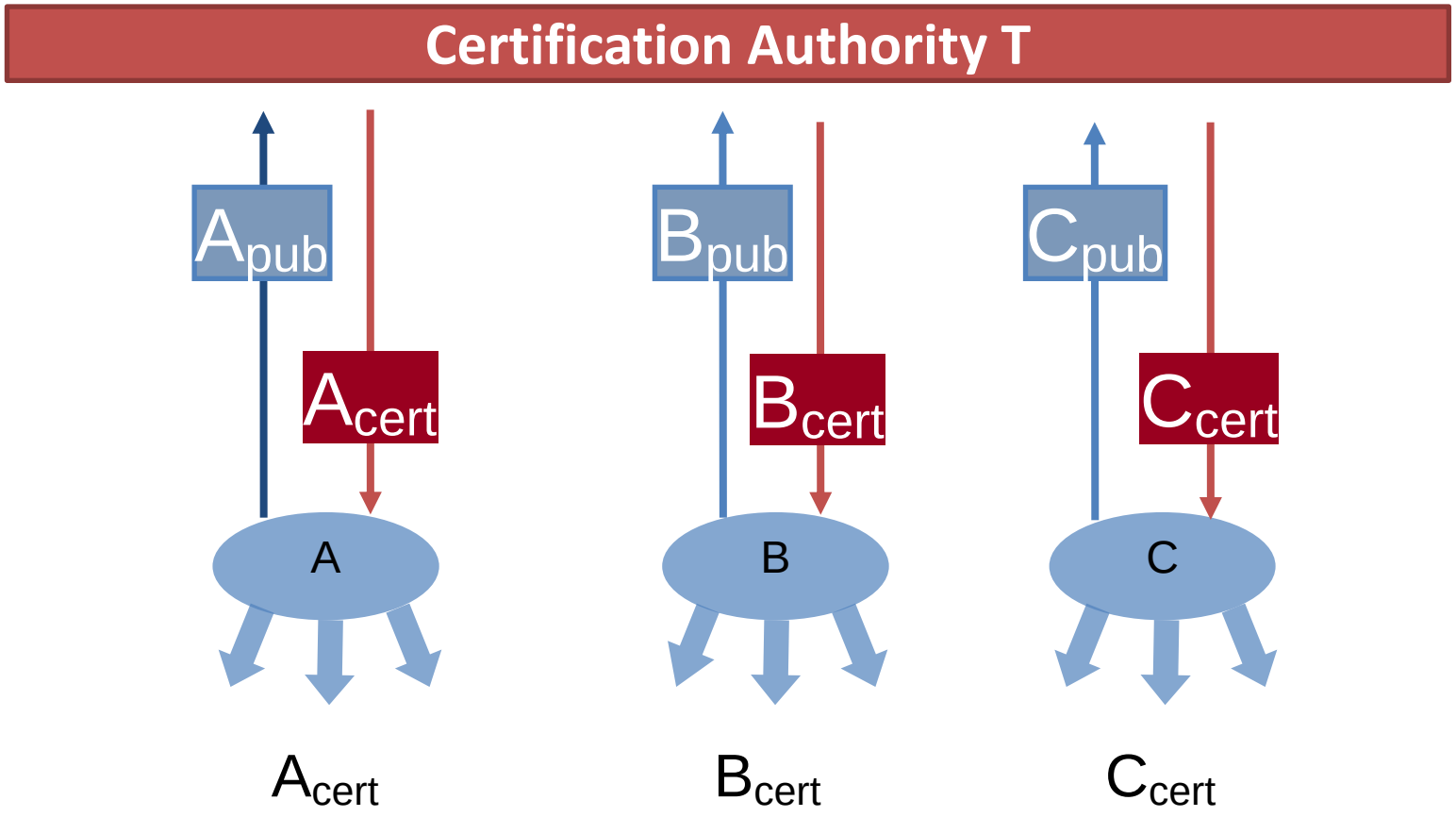
Public Key Dist. of Symmetric Keys

Take 3



Assumes Alice and Bob know each other's public key – How?

Certificates



$A_{cert} = \{A_{id}, A_{pub}, T_{expires}\} T_{sign}$

$A_{cert} = K^{-1}_{T} (A, K_A, T_{exp})$

Certificates

- A certificate is a statement that binds an identity to a cryptographic key
- The certificate is only as good as the steps taken by the certification authority to verify the identity and to protect the binding
- The public key of the certification authority is assumed to be known by the communicating parties
- What is the identifier? Does it need to be unique?
- Multiple certification authorities?

X. 509 Certificates

- Associate public key with user or service
- Signed by a certification authority (CA)
- Not forgeable – can be stored in a directory
- Issuer can revoke specific certificates – hold revocation list
- Certificate Fields:

Version of certificate format
Serial number – unique within CA

Signature algorithm

Issuer name + unique identifier

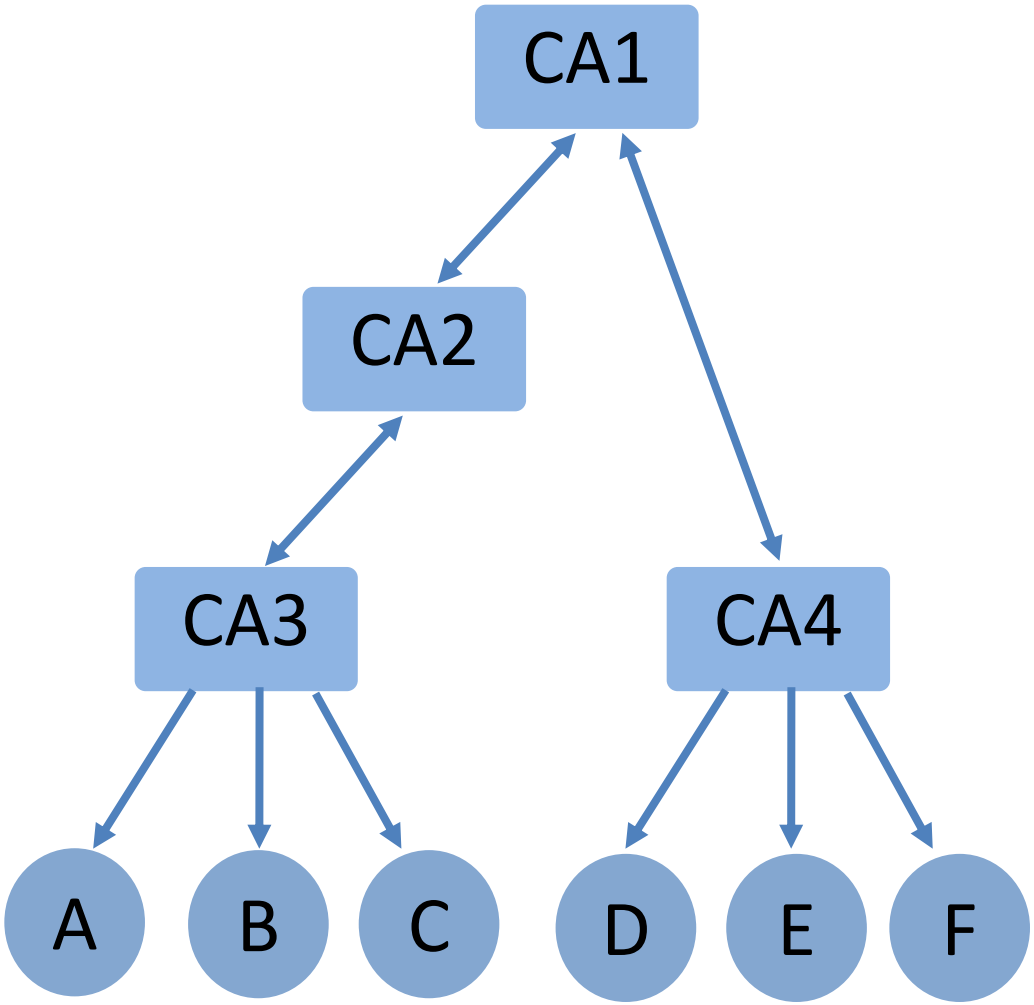
Period of validity

Subject name + unique identifier – to whom the certificate was issued

Extension fields for additional info about subject or issuer e.g. public key to be used to verify signature

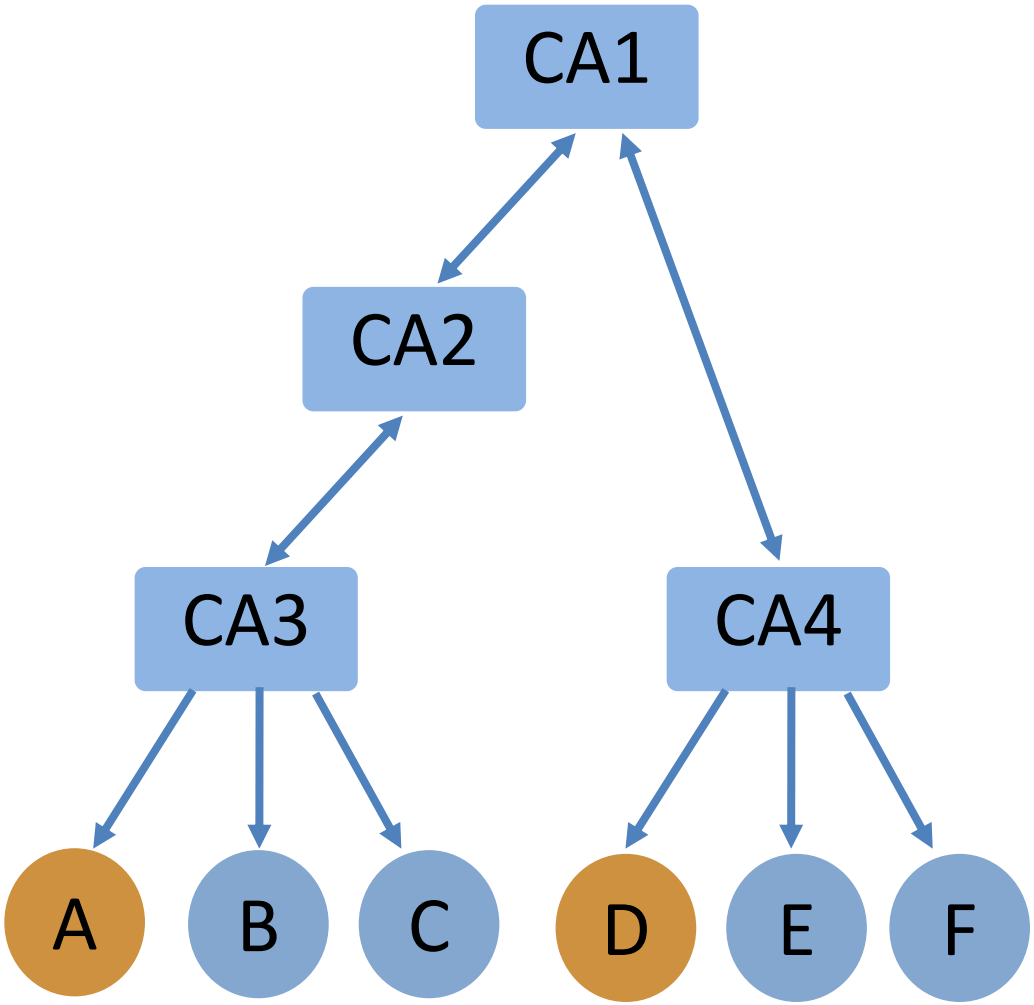
Signature – hash code of all other fields encrypted with issuer's private key

X.509 - Certification authority Hierarchy



- Each pair (parent-child) of CAs create a certificate for each other, e.g. CA1 for CA2 and vice-versa
- Certificates can be used to build a certification path

How can A verify D's certificate



A needs to verify $CA4_{sign}$ (D_{pub})

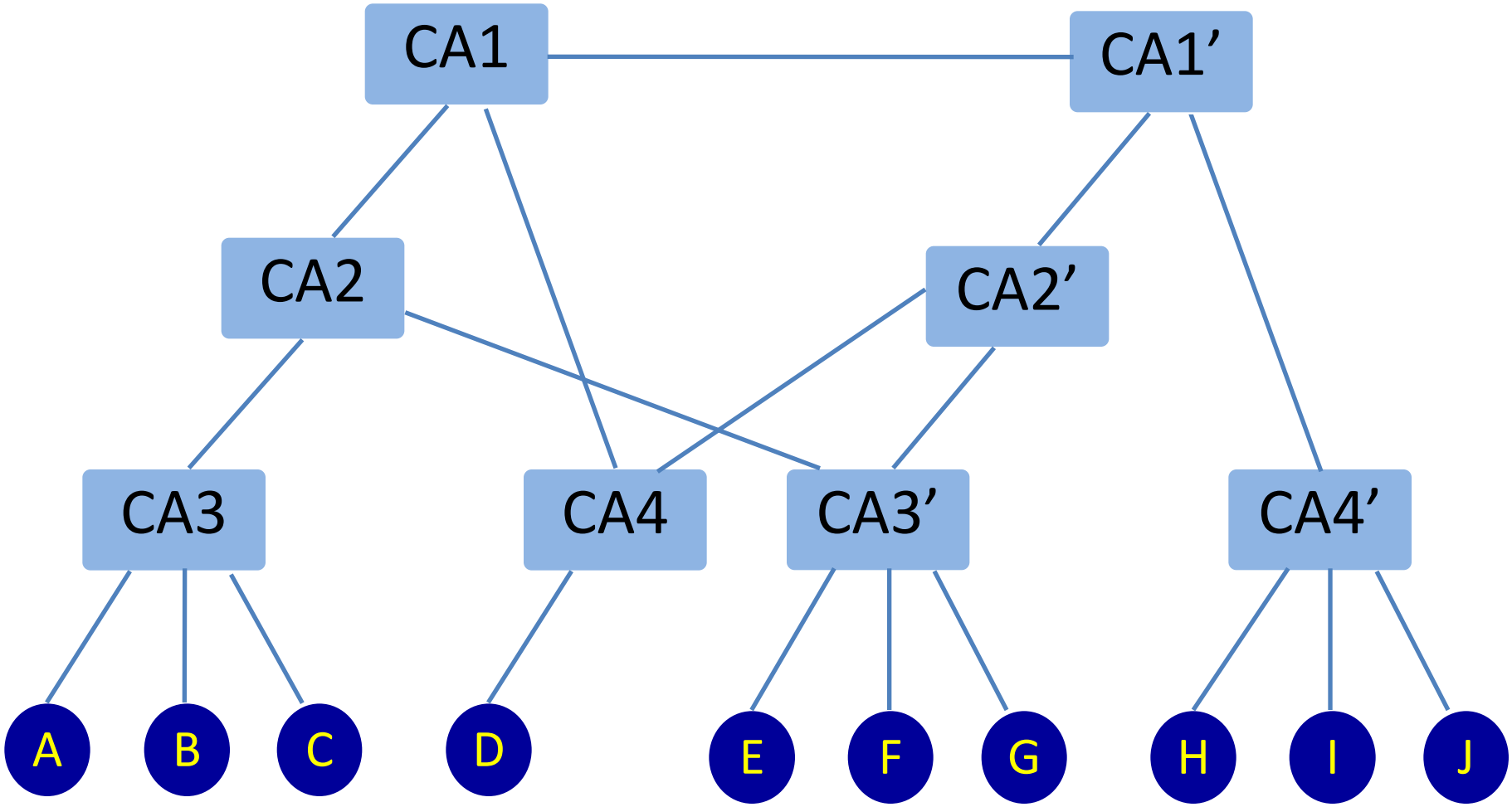
Gets from CA Hierarchy:

1) $CA3_{sign}$ ($CA2_{pub}$)
   $CA2_{sign}$ ($CA1_{pub}$)
   $CA1_{sign}$ ($CA4_{pub}$)

The certificates allow A to:

1) verify $CA2_{pub}$
2) verify $CA1_{pub}$
3) verify $CA4_{pub}$
4) verify D_{pub}

PKI Topologies



Summary

- Physical security impossible in Distributed Systems
- Networks inherently less secure than centralised computer systems as they are more susceptible to wire tapping etc.
- Cannot trust distributed workstations
- Dependence on distributed systems and value of information they manipulate make security an essential aspect

Summary

- Require
  - Authentication
  - Access Control
  - Data confidentiality and integrity Service
  - Notarisation Service
  - Notation for specification of Authorisation policy
- Encryption is an important mechanism for these services
- May be able to isolate critical components from network or use firewalls