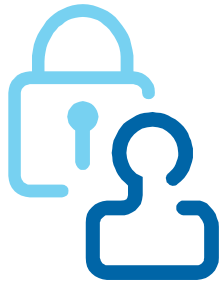
Block 3



EBU6010

Cryptography and Cyber Security

November 2023

Coursework details will be made available on QMPlus:

Deadline: BEFORE **4th December**

Dr Yasir Alfadhl BEng(Hons.) PhD FHEA SMIEEE

yasir.alfadhl@qmul.ac.uk

School of Electronic Engineering & Computer Science,

Queen Mary University of London

Topics to cover…

- **Key management**
  - **Key Exchange, Certificates & X.509**
- **Example of user Authentication/Access Control**
  - Kerberos
- **IPSec**
- **Firewalls**

Key Management

Key Exchange and Certificates

Classes of Keys

Lifetimes

- **Short term keys**
  - (ephemeral keys, session keys)
  - They are generated automatically.
  - Used for one message or session.
- **Long term keys**
  - Generated explicitly by the users.
  - They are used for
    - Authentication
    - Confidentiality (encryption)

- **Type of service**
  - Authentication keys
    - Public keys may have a long lifetime (decades)
    - Private keys / conventional keys have a shorter lifetime (year or two)
  - Confidentiality keys
    - Should have the shortest possible time.

Key Management Aspects

- **Distribution of keys**
- **Establishing a shared key with a third party.**
- **Key storage** (should be secure!)
- **Revocation of keys**

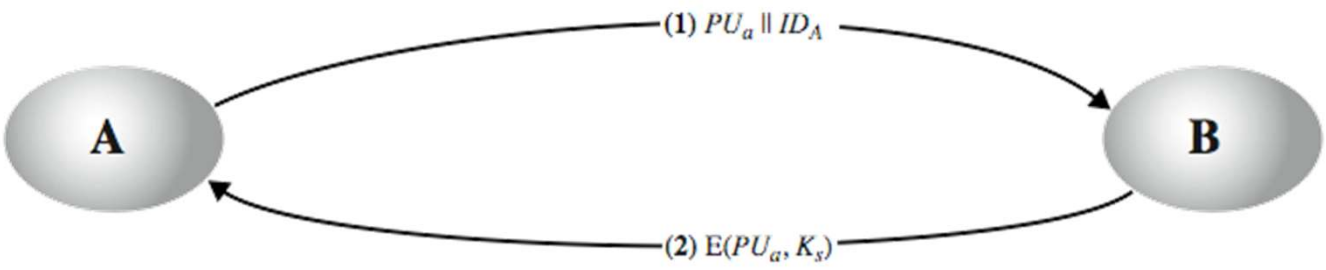
Key Distribution Issues

- Hierarchies of Key Distribution Centres (KDC's) required for large networks, but must trust each other.
- Session key lifetimes should be limited for greater security.
- Use of automatic key distribution on behalf of users, but must trust system.
- Use of de-centralised key distribution.
- Controlling key usage.

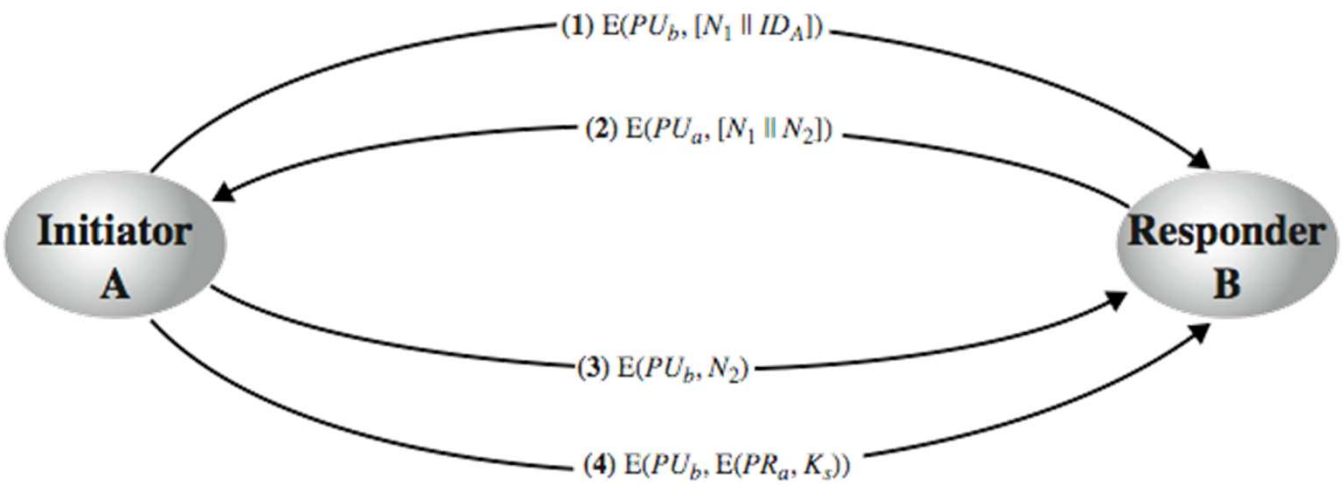
Simple Secret Key Distribution

– Allows secure communications

– No keys before/after exist



Secret Key Distribution with Confidentiality and Authentication



Hybrid Key Distribution

- Retains the use of private KDC.
- Shares a secret master key with each user.
- Distributes session keys using the master key.
- Public-key is used to distribute master keys.
  - Especially useful with widely distributed users.
- Rationale:
  - Performance.
  - Backward compatibility.

Distribution of Public Keys

- **Can be considered as using one of:**
  - Public announcement.
  - Publicly available directory.
  - Public-key authority.
  - Public-key certificates.

Public Announcement

- Users distribute public keys to recipients or broadcast to community at large;
  - eg. append PGP keys to email messages or post to news groups or email list.

- Major weakness is forgery:
  - Anyone can create a key claiming to be someone else and broadcast it.
  - Until forgery is discovered can masquerade as claimed user.

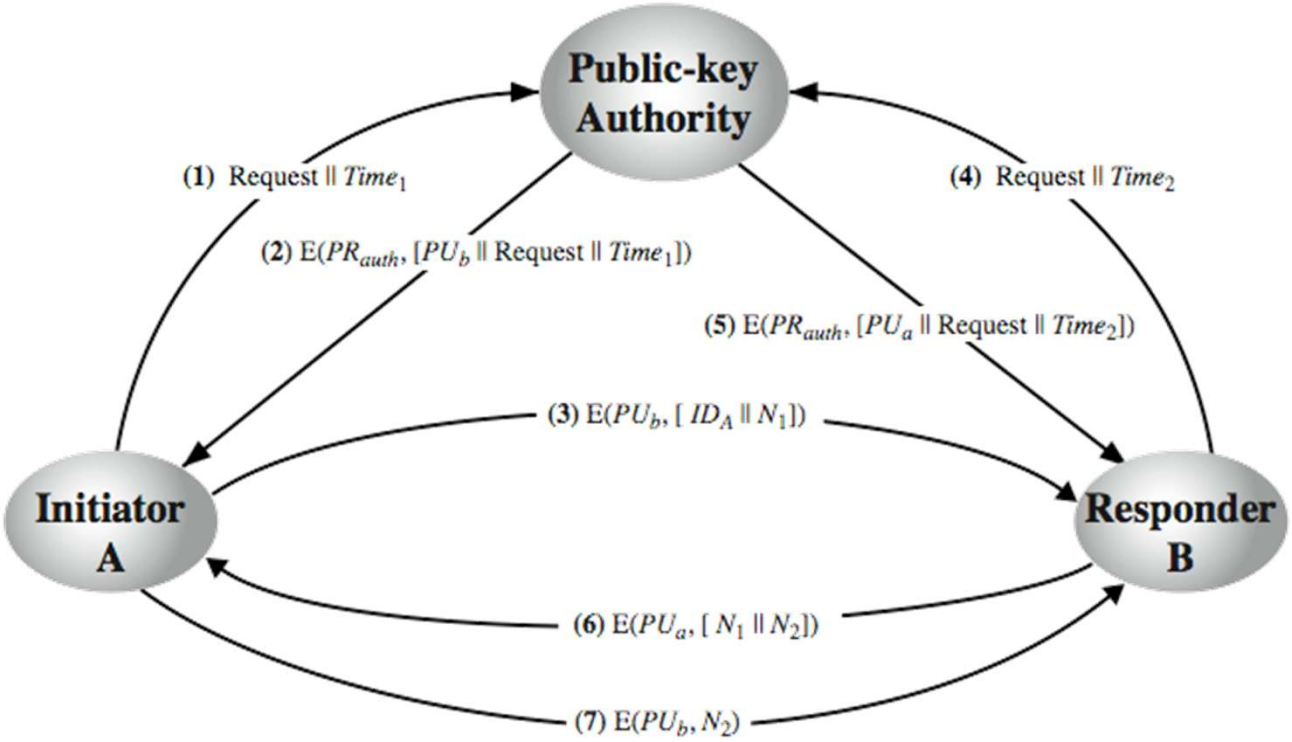
Publicly Available Directory

- Can obtain greater security by registering keys with a public directory.
- Directory must be trusted with properties:
  - Contains {name, public-key} entries.
  - Participants register securely with directory.
  - Participants can replace key at any time.
  - Directory is periodically published.
  - Directory can be accessed electronically.
- Still vulnerable to tampering or forgery.

Public-Key Authority

- Improves security by tightening control over distribution of keys from directory.
- Has properties of directory and requires users to know public key for the directory;
- Users interact with directory to obtain any desired public key securely:
  - Requires real-time access to directory when keys are needed.
  - May be vulnerable to tampering.

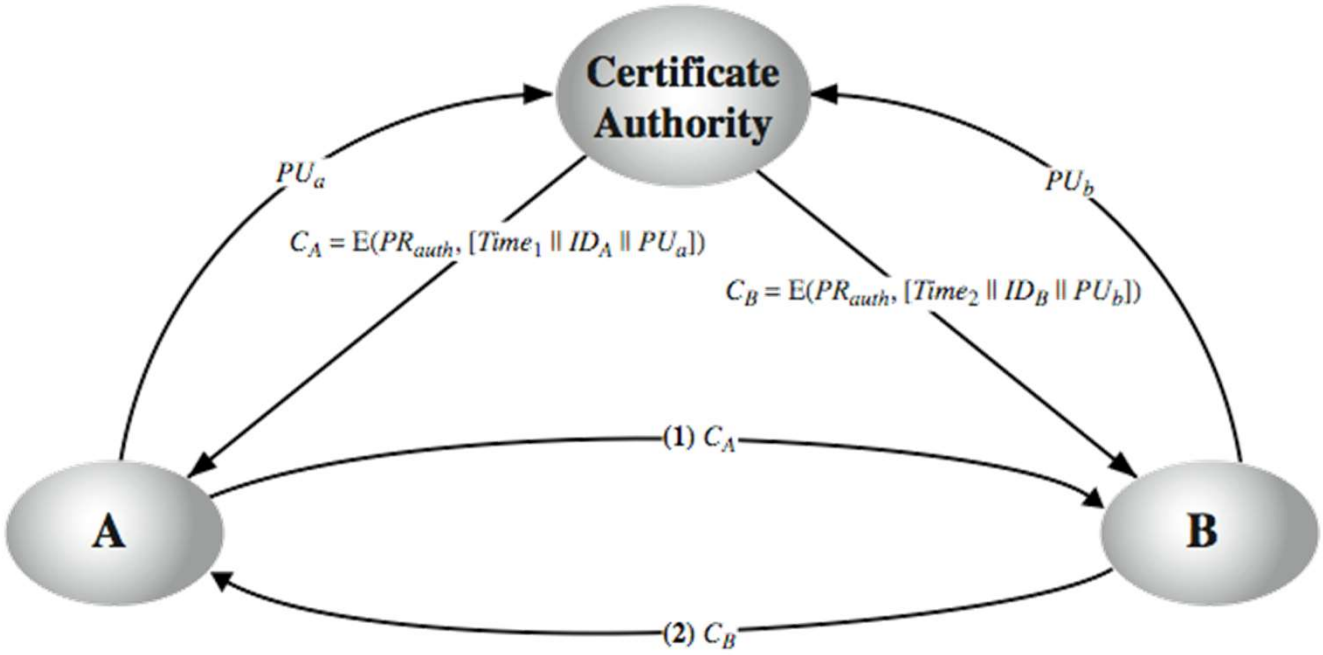
Public-Key Authority



Public-Key Certificates

- Certificates allow key exchange without real-time access to public-key authority.
- A certificate binds **identity** to **public key**.
  - Usually with other info such as period of validity, rights of use etc.
- With all contents **signed** by a trusted Public-Key or Certificate Authority (CA).
- Can be verified by anyone who knows the public-key authorities public-key.

Public-Key Certificates



Summary

- Symmetric key distribution using public-key encryption.
  - RSA, Diffie-Hellman, ..
- Distribution of public keys.
  - Announcement, directory, authority, CA.

X.509

Public Key Infrastructure (PKI)

CA and X.500 history

- X.500 is a series of recommendations that define a directory service. The directory is distributed in several servers.
- X.500 introduced the Distinguished Name (DN), a guaranteed unique name for everyone on earth. (typical DN component, Country, State, Locality, Organisation, Common Name...)
- Because of concerns about misuse of the X.500 directory certificates were intended to protect access to the directory.

X.509 Authentication Service

- Part of X.500 directory service standards.
- **Defines framework for authentication services.**
  - Directory may store public-key certificates.
  - With public key of user signed by certification authority.
- Also defines authentication protocols.
- **X.509 is based in public-key certificates and digital signatures (used S/MIME, IP security, SSL/TLS, SET). X.509 does not dictate which public-key algorithm to use but recommends RSA.**
- X.509 certificates are widely used.
  - have 3 versions.

X.509

- Certificates are placed in the directory by a CA or by the user (not by the directory server)
  - Any user with access to the public key of the CA can recover the user public key that was certified.
  - Only the certification authority can modify the certificate. Any modification by a third party will be detected.

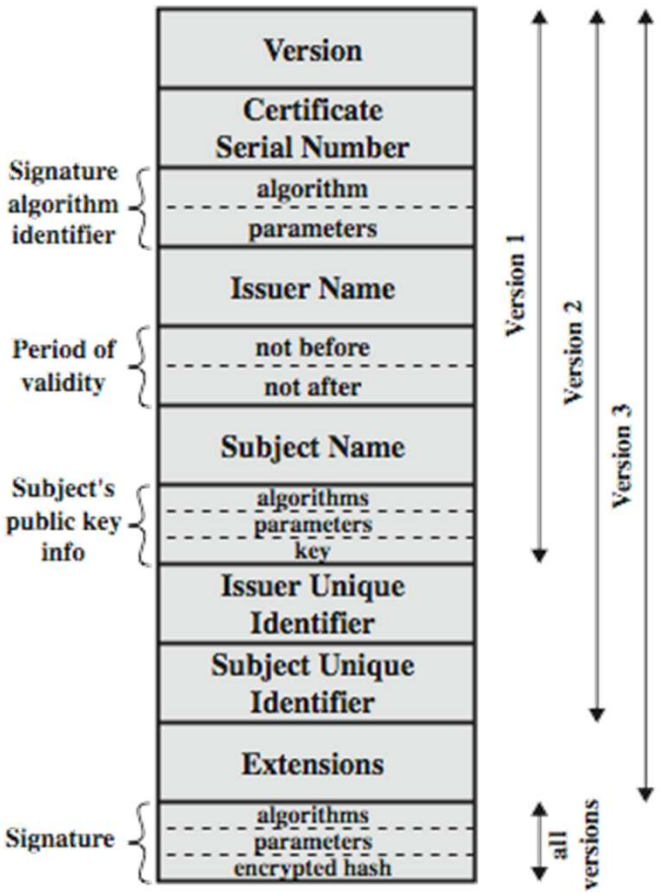
X.509

The certificate contains:

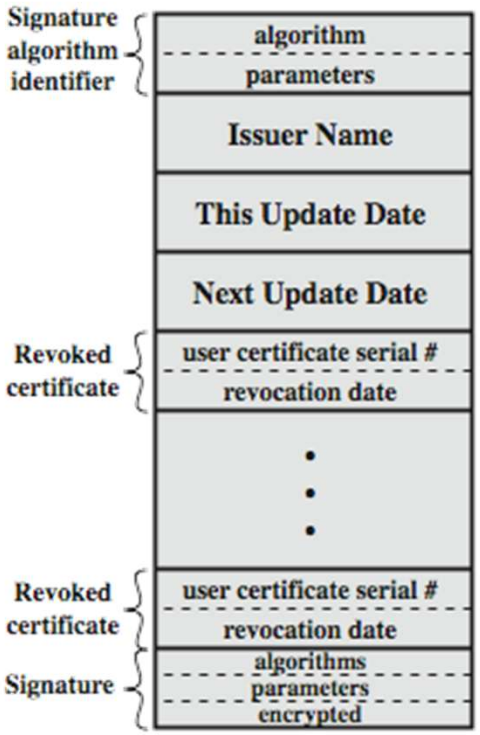
Version (V)	(**1**, **2**, or **3**)
Serial Number (SN)	Unique within CA -- identifying certificate
Signature algorithm identifier (AI)	
Issuer X.500 name (CA)	
Period of validity (TA)	From - to dates
Subject X.500 name (A)	Name of owner
Subject public-key info (Ap)	algorithm, parameters, key
Issuer unique identifier	v2+
Subject unique identifier	v2+
Extension fields	V3
Signature	.. of hash of all fields in certificate.

- Notation CA<<A>> denotes certificate for A signed by CA

X.509 Certificates Formats



(a) X.509 Certificate

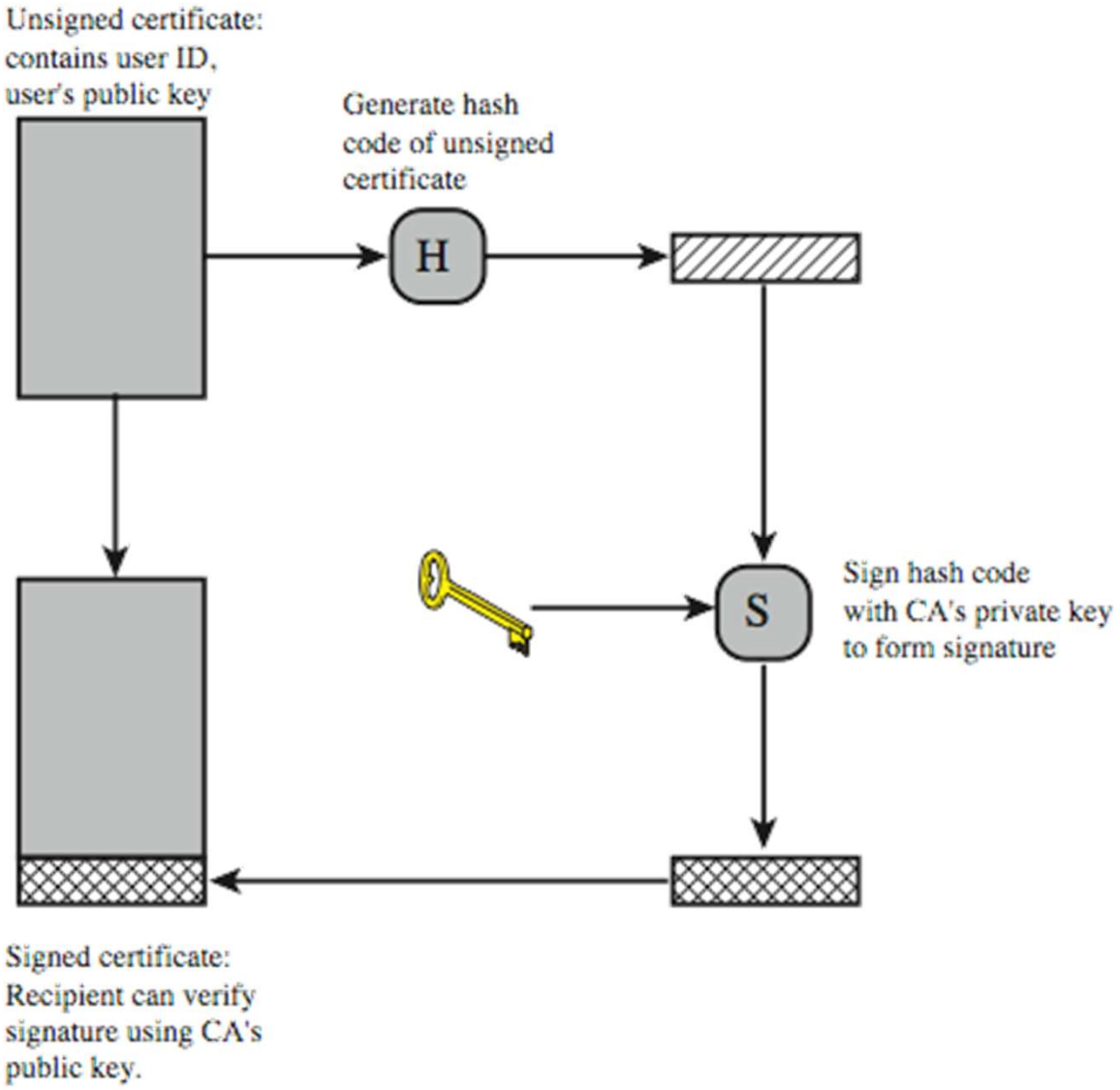


(b) Certificate Revocation List

X.509 Version 3

- Additional information fields added in the certificate.
  - Email/URL, policy details, usage constraints.
- Rather than explicitly naming new fields defined a general extension method.
- Extensions consist of:
  - Extension identifier.
  - Criticality indicator.
  - Extension value.

X.509 Certificate Use



Revocation

- Need revocation if:
  - User's Private-Key has been compromised
  - Certification Authority has been compromised
  - User is no longer certified by this Authority
- Certificate Revocation List (CRL)
- Users should check certificates with CA's CRL.

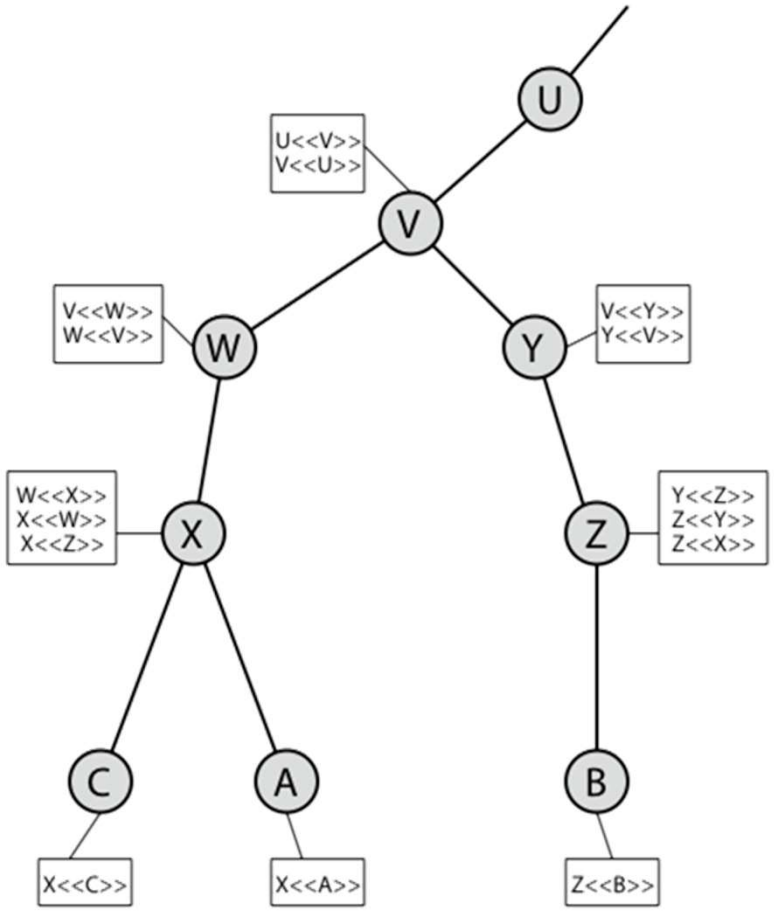
Authentication Procedures

- If A and B were to exchange information using two keys issued from different Certification Authorities (CAs)?
  - In X509:
    - User A sends B a key, which was issued by CA_A
    - User B receives it, but does not know (or trust) CA_A, hence requests CA_B to supply a certificate of CA_A.
    - CA_B supplies B with a certificate of CA_A, signed with CA_B.
  - This is applied in exchange of emails, websites, etc.
  - You might have noticed Internet browsers sometime ask for your judgement to permission to enter a website as the certificate is not recognised!

CA Hierarchy

- If both users share a common CA then they are assumed to know its public key.
- Otherwise CA's must form a hierarchy.
- Use certificates linking members of hierarchy to validate other CA's.
  - Each CA has certificates for clients (forward) and parent (backward).
- Each client trusts parents certificates.
- Enable verification of any certificate from one CA by users of all other CAs in hierarchy.

CA Hierarchy Use



"the use of an X.509 hierarchy to mutually verify clients certificates"

User Authentication

User Authentication

- Is the process of verifying an identity claimed by or for a system entity.

- **Has two steps**:
  - Identification - specify identifier
  - Verification - bind entity (person) and identifier

- Distinct from message authentication

Authentication Protocols

- Used to convince parties of each other's identity and to exchange session keys.
- May be one-way or mutual.
- Key issues are:
  - Confidentiality – to protect session keys.
  - Timeliness – to prevent replay attacks.

Identity Management

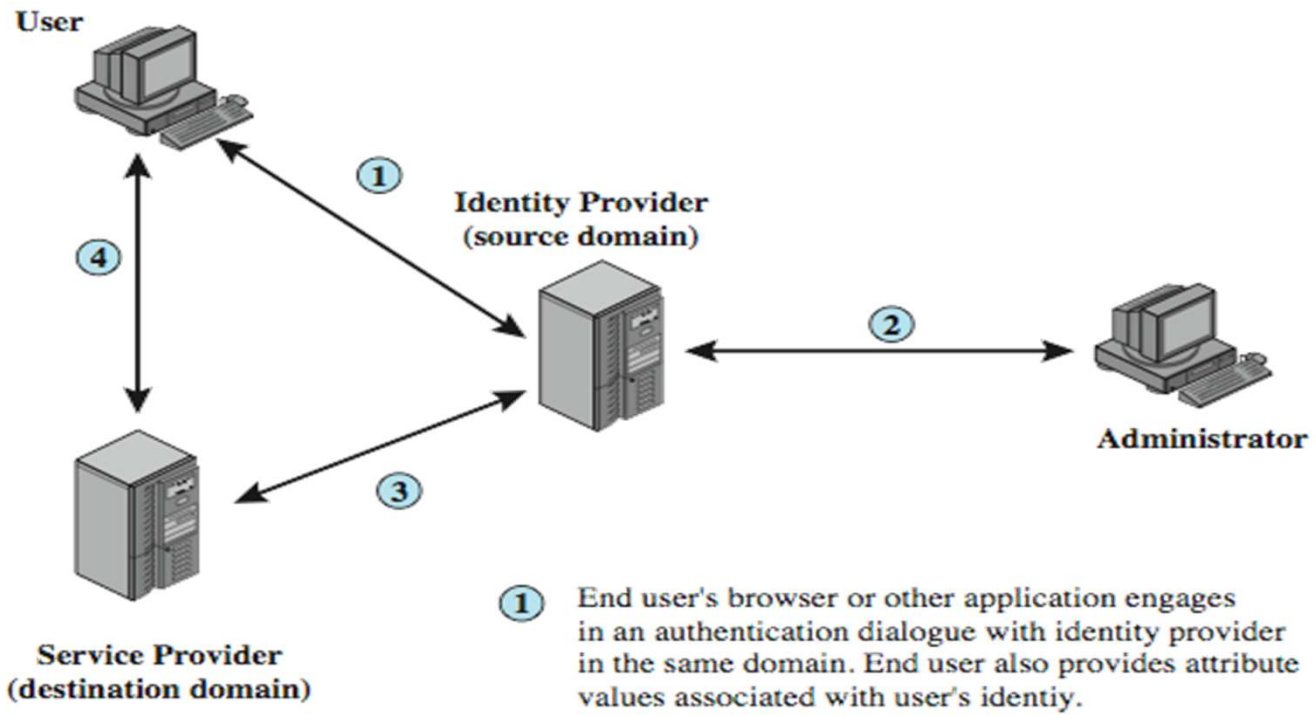
Federated Identity Management

- Use of common identity management scheme
  - Across multiple enterprises & numerous applications
  - Supporting many thousands, even millions of users

- Principal elements are:
  - Authentication, authorisation, accounting, provisioning, workflow automation, delegated administration, password synchronisation, self-service password reset, federation

- Kerberos contains many of these elements

Identity Federation



1. End user's browser or other application engages in an authentication dialogue with identity provider in the same domain. End user also provides attribute values associated with user's identiy.
2. Some attributes associated with an identity, such as allowable roles, may be provided by an administrator in the same domain.
3. A service provider in a remote domain, which the user wishes to access, obtains identity information, authentication information, and associated attributes from the identity provider in the source domain.
4. Service provider opens session with remote user and enforces access control restrictions based on user's identity and attributes.

Standards Used

- Security Assertion Markup Language (SAML)
  - XML-based language for exchange of security information between online business partners
- Part of OASIS (Organisation for the Advancement of Structured Information Standards) standards for federated identity management
  - e.g. WS-Federation for browser-based federation
- Need a few mature industry standards

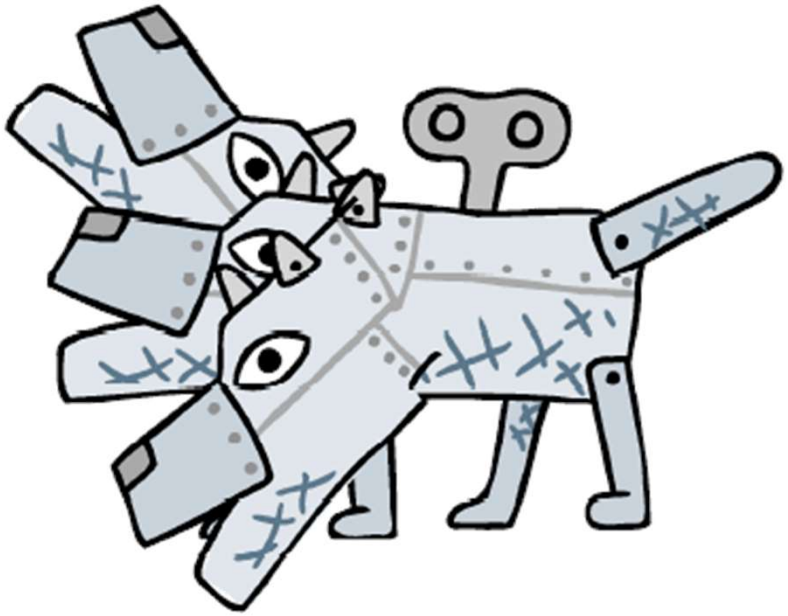
Summary

- Kerberos
  - **Definition:** Kerberos is an authentication service designed for use in a distributed environment.
  - **Uses:** It makes use of a trusted third-party authentication service that enables clients and servers to establish authenticated communication.

- Federated Identity Management is a relatively new concept dealing with the use of a common identity management scheme across multiple enterprises and numerous applications and supporting many thousands, even millions of users.
  - Kerberos contains many of these elements.

Example of a User Authentication and Access Control Application

- Kerberos



Kerberos

- Kerberos is an authentication service which addresses the following problem:

In an open distributed environment in which users want to access services on servers distributed throughout the network, how can the servers:

  - Restrict the access to authorised users, and
  - Authenticate requests for service.

Threats

- **Impersonation:** A user gains access to a particular workstation and pretends to be another user operating from that workstation.
- **Workstation "Impersonation":** A user modifies the network address of a workstation so that the request sent by the modified workstation appear to be from the "impersonated" workstation.
- **A user eavesdrop on information exchanges and use a replay attack to gain entrance to a server** (or disrupt operations).

Kerberos Approach

- Kerberos provides a *centralised* authentication server that authenticates
  - User to servers
  - Servers to users

- Network Authentication Protocol
- Developed at MIT in the mid 1980s.
- Available as open source or in supported commercial software.

Kerberos Approach

- Kerberos uses *conventional* encryption (DES). There are two versions of Kerberos:
  - Version 4: Is the 'original' Kerberos as versions 1-3 were internal development versions. This version still exists in many applications.
  - Version 5: Corrects some of the security deficiencies found in version 4.

Motivation

- For the authentication of dedicated user workstations (clients) and distributed or centralised servers we can do the following:
  - Rely on clients to assure identify of its user(s) and rely on each server to enforce the security policies (based on ID).
  - Require that client systems authenticate themselves to servers. Trust the client system concerning the identity of the user.
  - Require the user to prove identity for each service invoked. Require that servers prove their identity to clients (The Kerberos Approach).

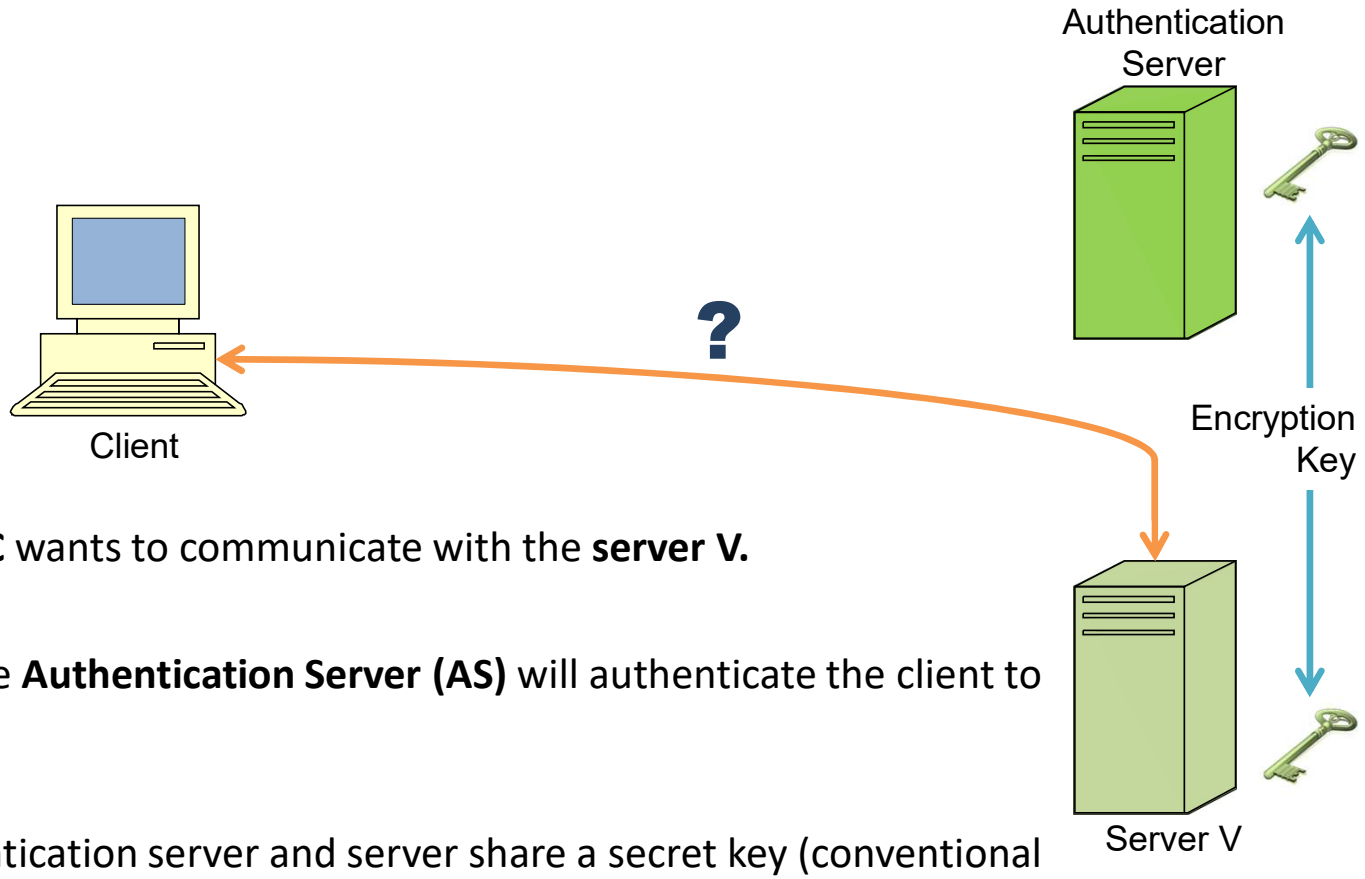
Requirements

- **Secure:** Should cope with external attacks.
- **Reliable:** Kerberos should be highly reliable and should employ distributed server architecture. One system able to back up another.
- **Transparent:** Apart from the password, a user should not be aware of the authentication is taking place.
- **Scalable:** As the network grows, there should be a method which allows such growth.

Simple Authentication

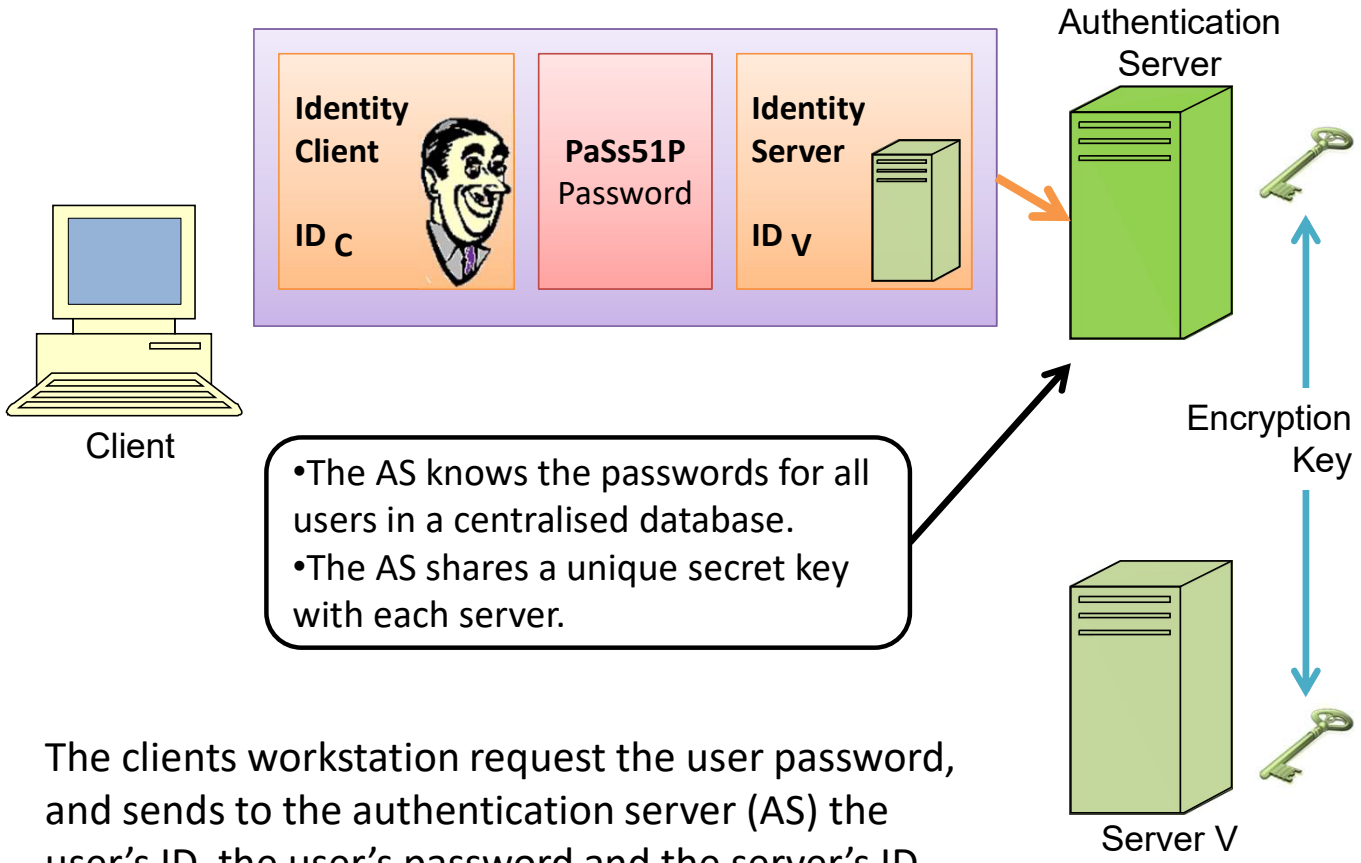
- A simple authentication is when a new student wants to use the university library. Before it can use the library
  - The student identifies himself/herself with the university administration.
  - The administration check the students details. If a valid student, then it provides him/her with a student ID card.
  - The student goes to the library and shows this ID card so she/h has access to the library.

Simple Authentication



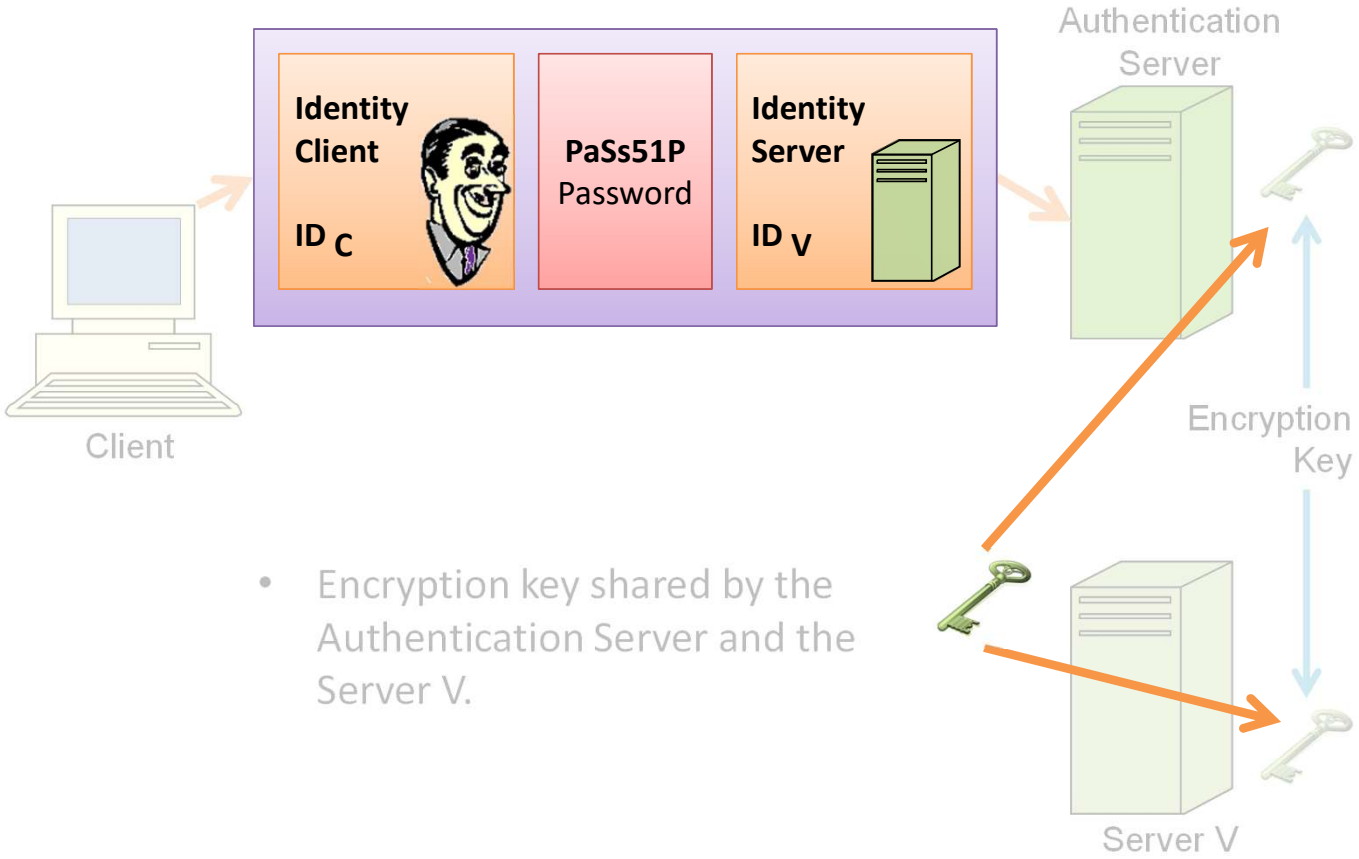
- The **client C** wants to communicate with the **server V.**
- To do so the **Authentication Server (AS)** will authenticate the client to the server.
- The Authentication server and server share a secret key (conventional encryption)

Simple Authentication



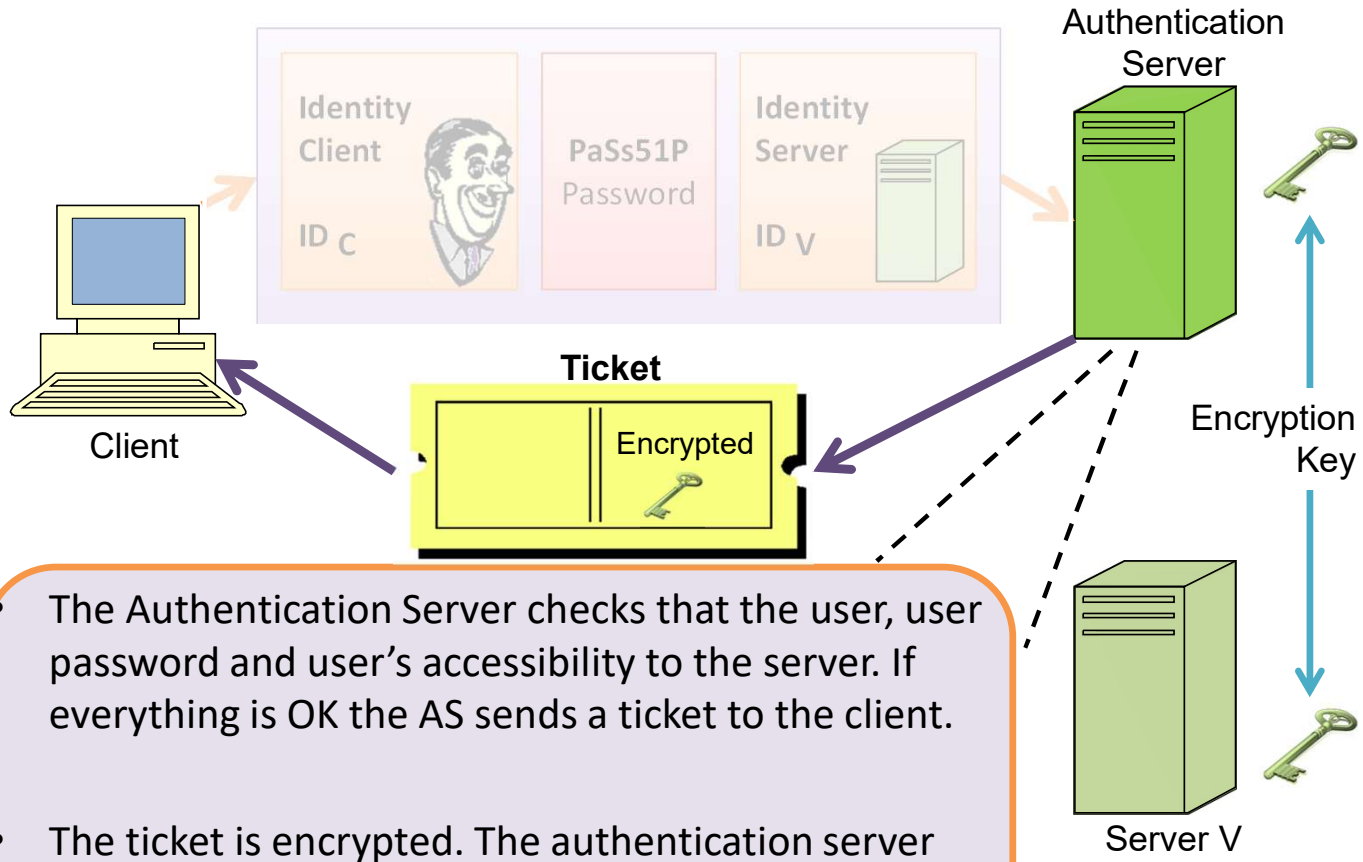
- The clients workstation request the user password, and sends to the authentication server (AS) the user's ID, the user's password and the server's ID.

Simple Authentication



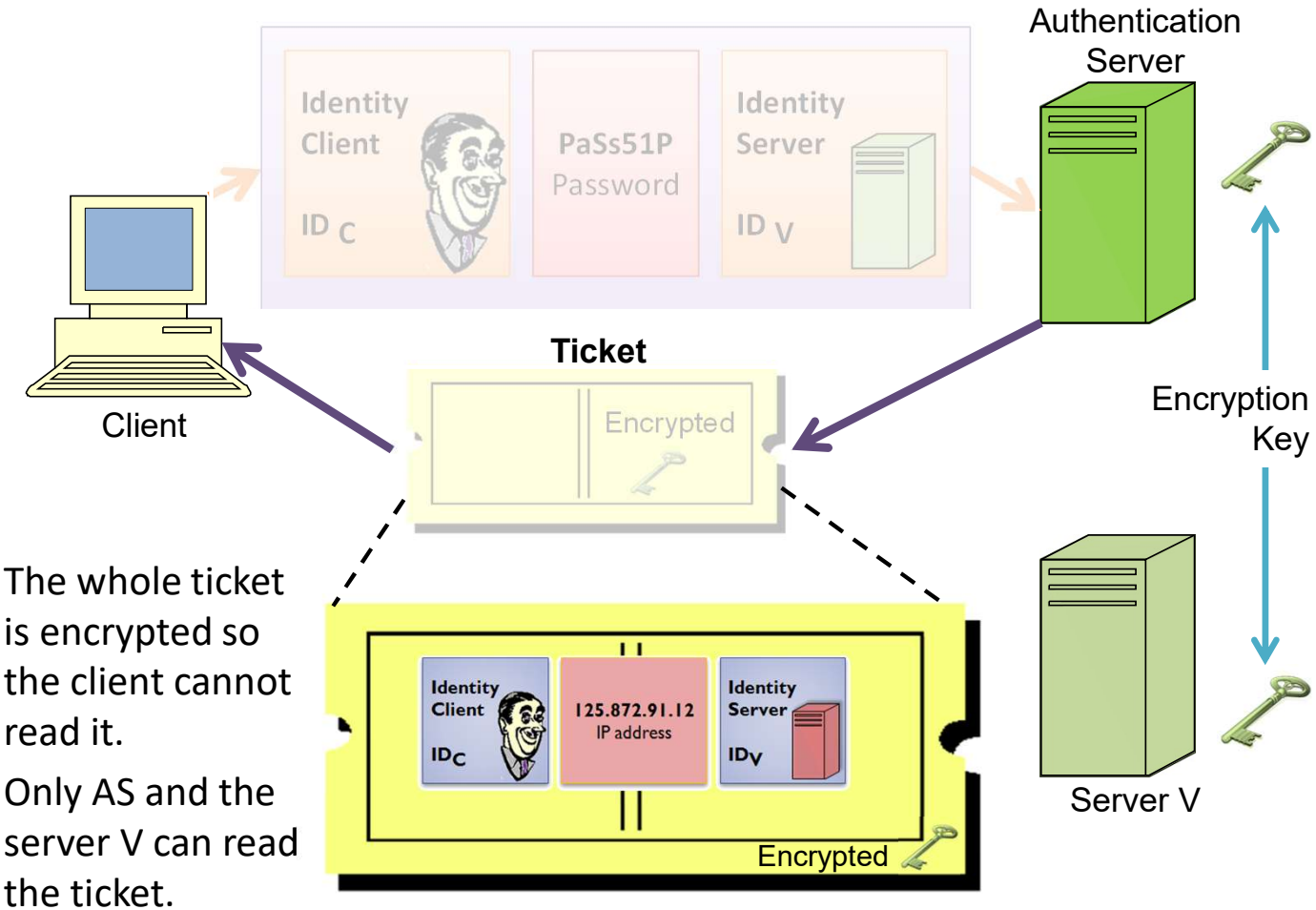
- Encryption key shared by the Authentication Server and the Server V.

Simple Authentication



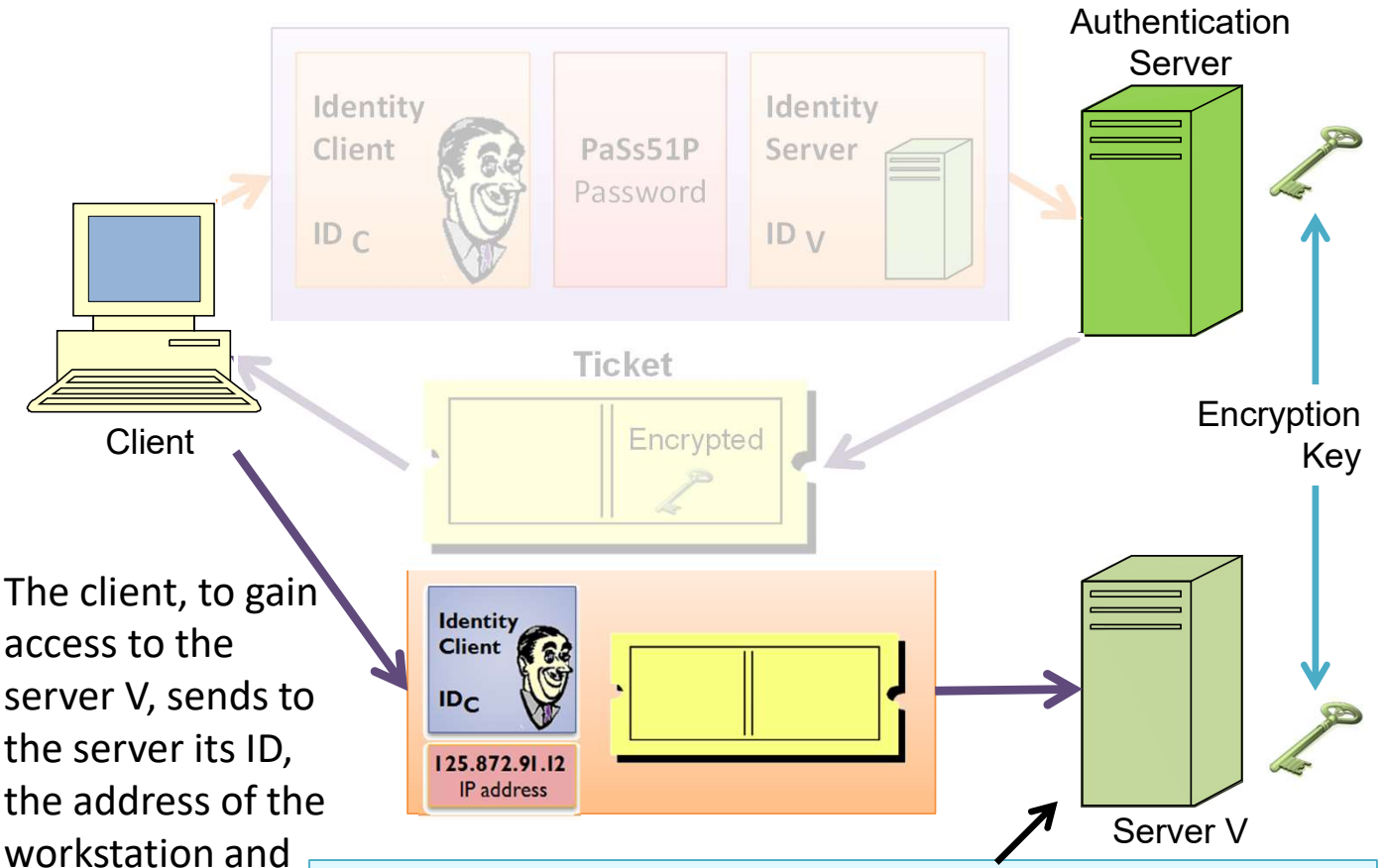
- The Authentication Server checks that the user, user password and user's accessibility to the server. If everything is OK the AS sends a ticket to the client.
- The ticket is encrypted. The authentication server and the server have a copy of the encryption key (not the client)

Simple Authentication



- The whole ticket is encrypted so the client cannot read it.
- Only AS and the server V can read the ticket.

Simple Authentication



- The client, to gain access to the server V, sends to the server its ID, the address of the workstation and the ticket.

The server decrypts the ticket and compares the contents of the ticket with the user's ID and network address that the client sent.

Simple Authentication Dialogue

(1) C → AS: $ID_C \| P_C \| ID_V$

(2) AS → C: $Ticket$

(3) C → V: $ID_C \| Ticket$

$Ticket = E(K_v, [ID_C \| AD_C \| ID_V])$

where

C = client

AS = authentication server

V = server

ID_C = identifier of user on C

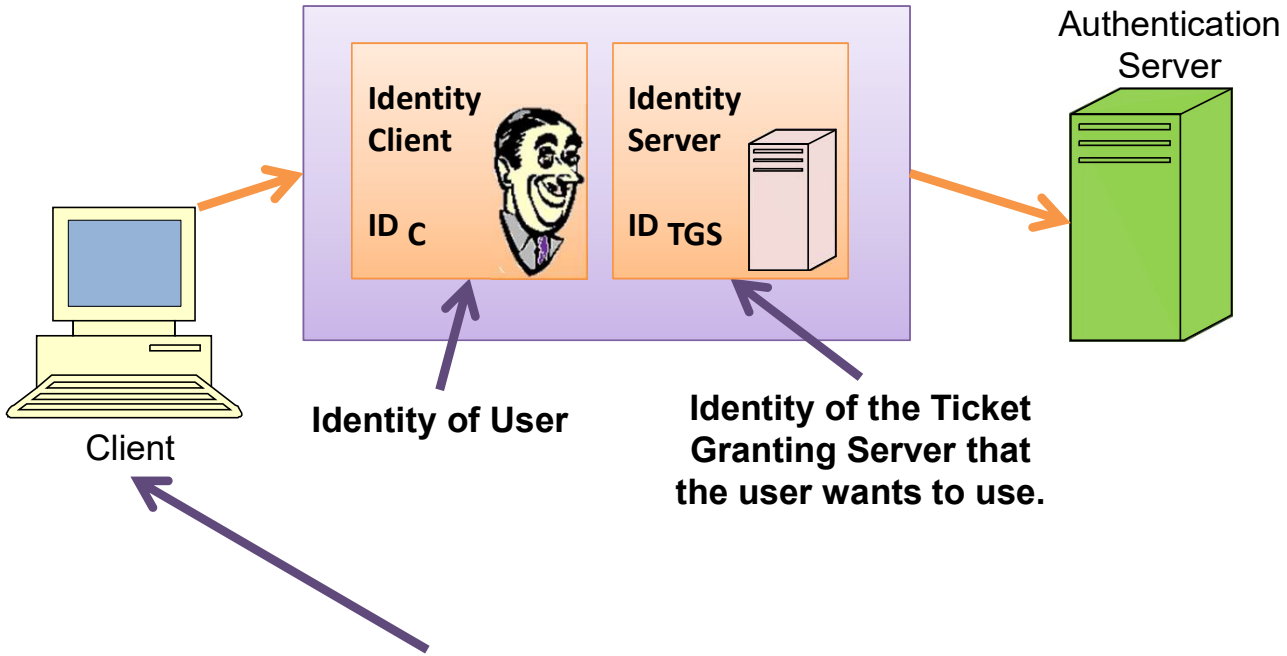
Problems

- Try to minimise the number of times a user enters a password.
- Plaintext transmission of the password.

More Secure Authentication

- To avoid transmitting the password as plaintext we introduce the Ticket Granting Server (TGS).

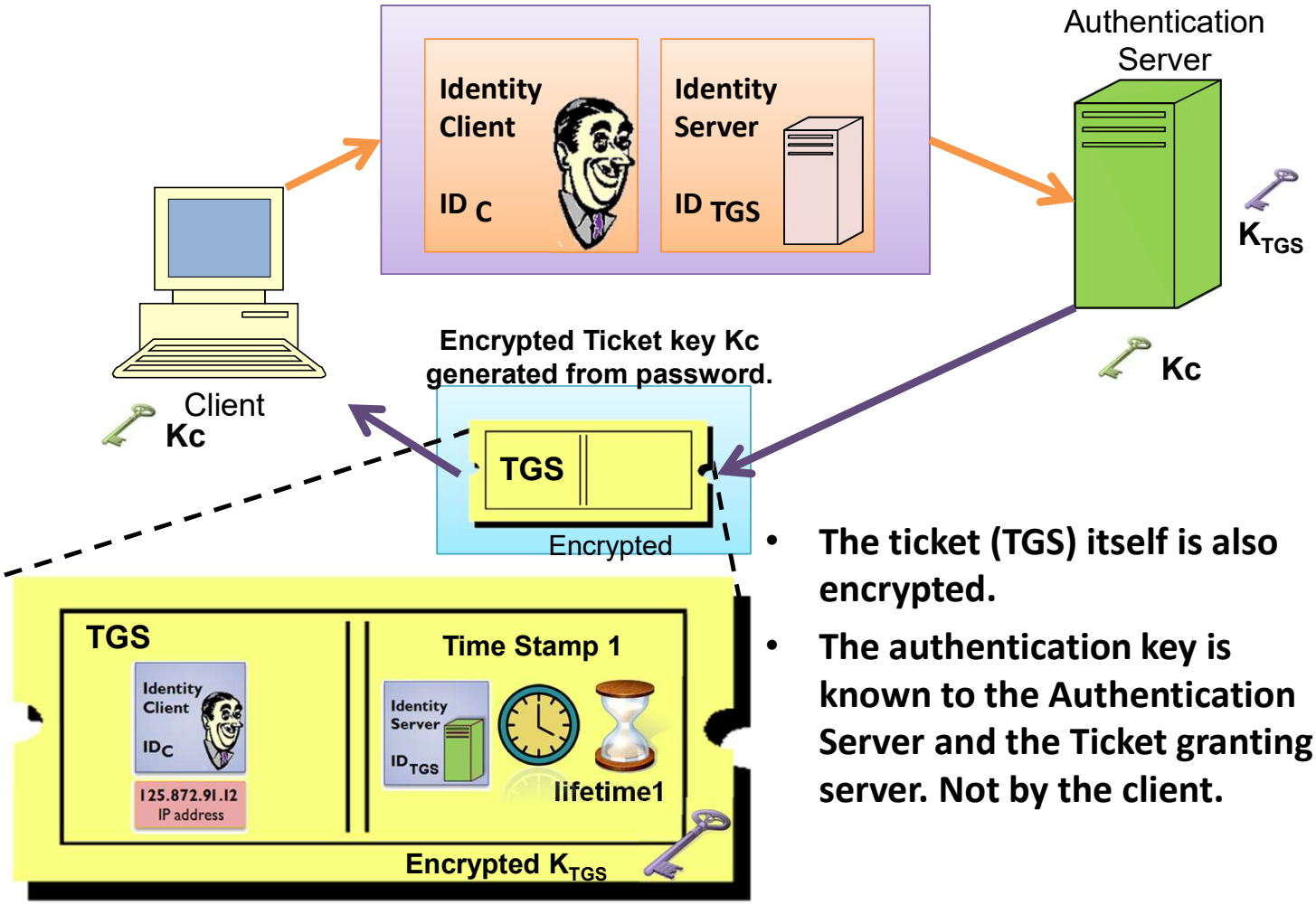
More Secure Authentication



- The client request from the authentication server (AS) a ticket- granting ticket on behalf of the user.

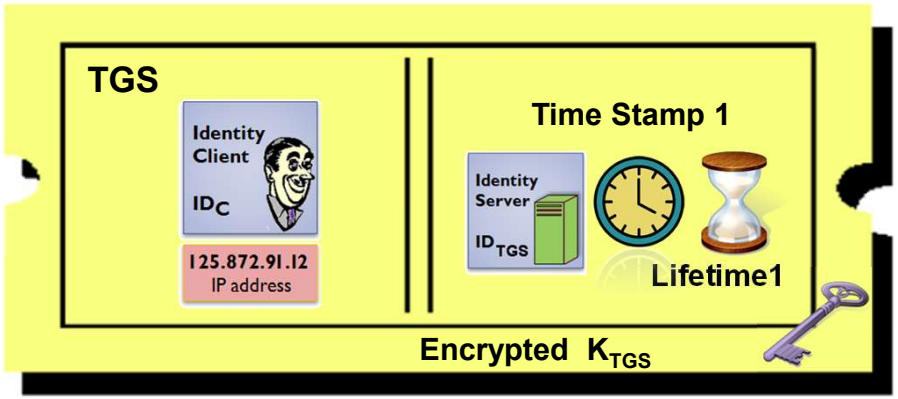
Once per user logon

More Secure Authentication



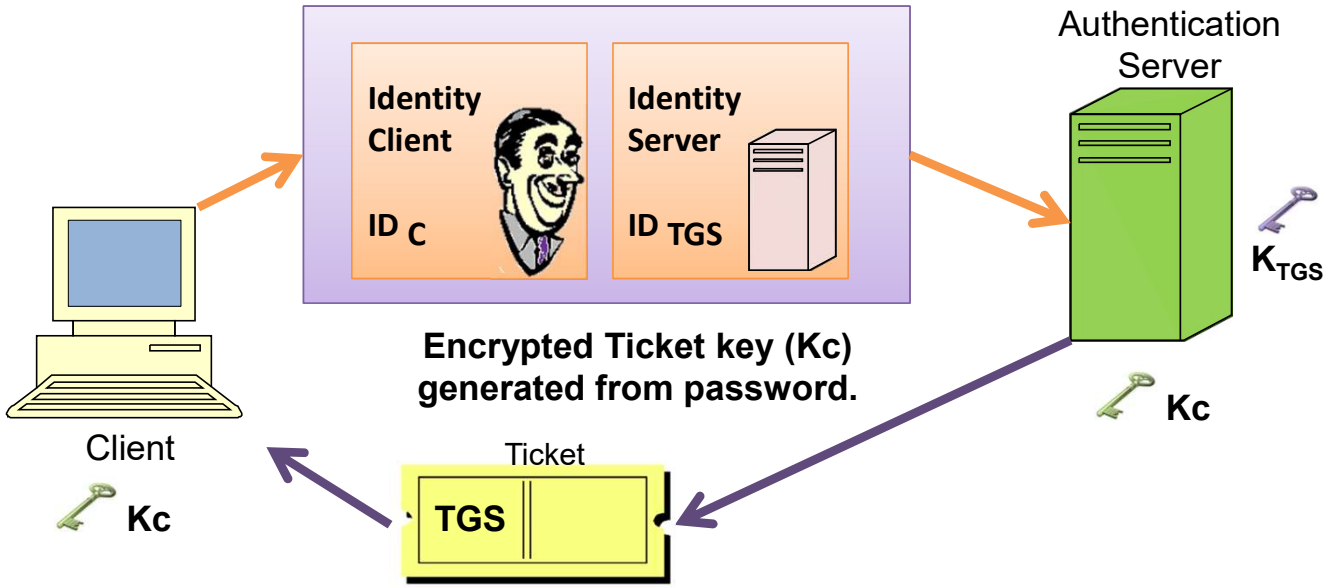
- The ticket (TGS) itself is also encrypted.
- The authentication key is known to the Authentication Server and the Ticket granting server. Not by the client.

More Secure Authentication



- AS sends the ticket-granting service (TGS) ticket, the whole ticket is double encrypted. First with using a key known to AS and the ticket granting server, then using a key generated by the password.
- The encryption key Kc is generated from the password of the user (AS knows the password). Notice that the password never transmitted.
- The ticket contains an IP address, a time stamp and a lifetime.

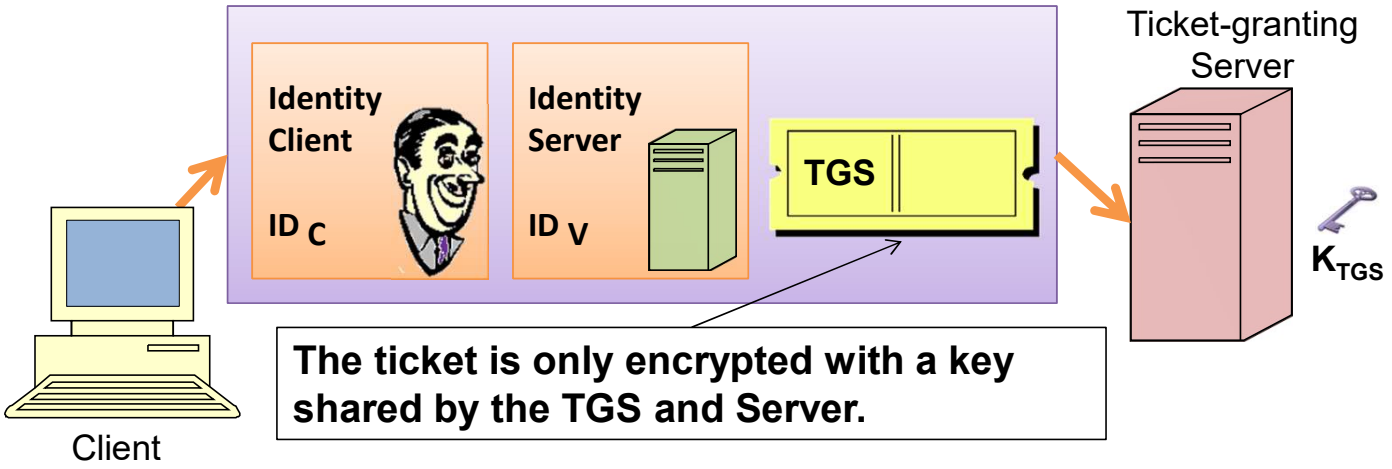
More Secure Authentication



- **The client ask the users for his/her password, generates the key Kc, and recovers the TGS ticket.**

- **The authentication between AS and User is done using Kc.**

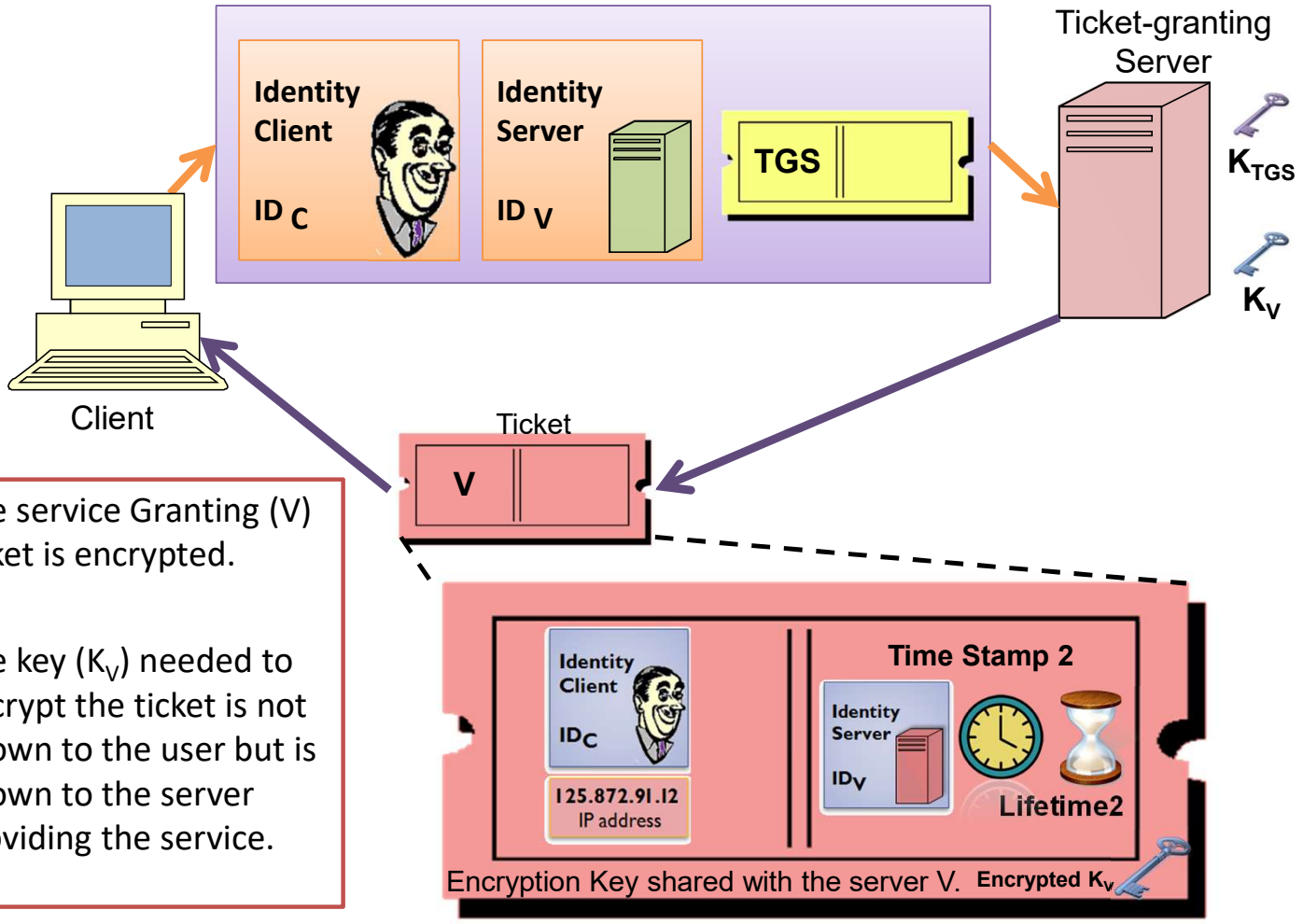
Type of Service



- The client request from Ticket Granting Service a service-granting ticket.
- It sends the user's ID, the desire service ID and the ticket-granting ticket.
- The TGS decrypts the ticket-granting ticket (using K_{TGS}). Authenticates the user (comparing the incoming information and ticket information) and checks if the user has access to the requested server. TGS also checks if the lifetime has not expired.

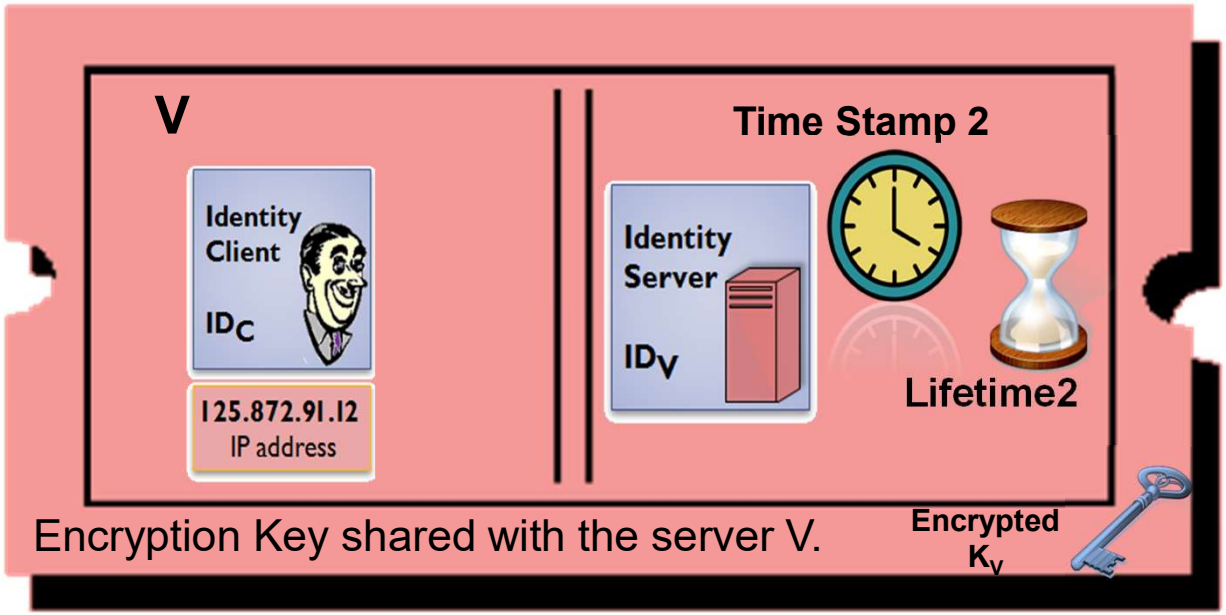
Once per type of service

Type of Service



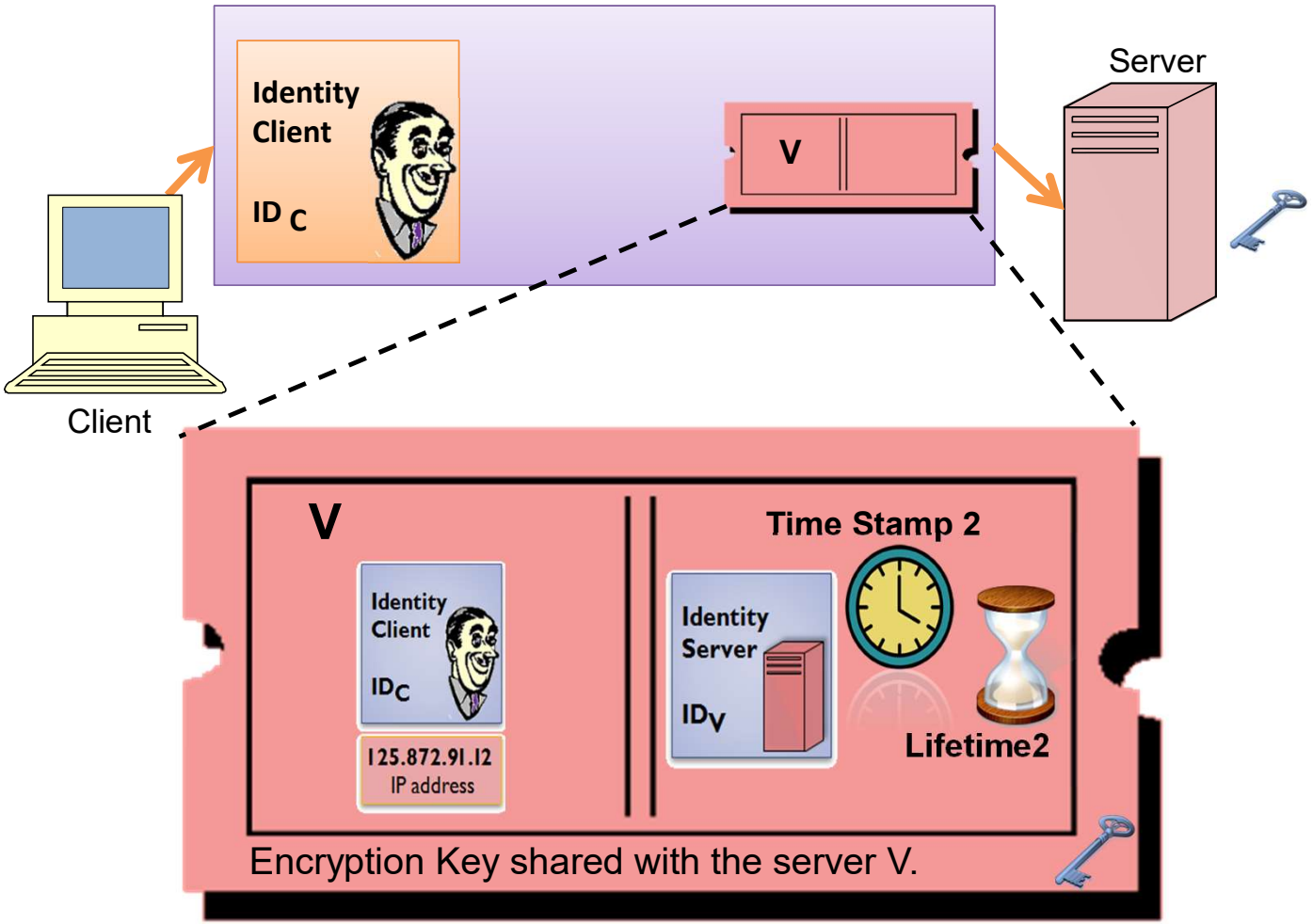
- The service Granting (V) ticket is encrypted.
- The key (K_V) needed to decrypt the ticket is not known to the user but is known to the server providing the service.

Type of Service



- The service-granting ticket (V) is encrypted. The encryption key K_V is known by the service granting server and the server.
- The ticket contains a time stamp and lifetime.

Session


Identity
Client
ID C
V
Server
Client
V
Time Stamp 2
Identity
Client
IDC
125.872.91.12
IP address
Identity
Server
IDV
Lifetime2
Encryption Key shared with the server V.

More Secure Authentication Dialogue

Once per user logon session:

(1) $C \rightarrow AS: \quad ID_C \| ID_{tgs}$

(2) $AS \rightarrow C: \quad E(K_c, Ticket_{tgs})$

Once per type of service:

(3) $C \rightarrow TGS: \quad ID_C \| ID_V \| Ticket_{tgs}$

(4) $TGS \rightarrow C: \quad Ticket_v$

Once per service session:

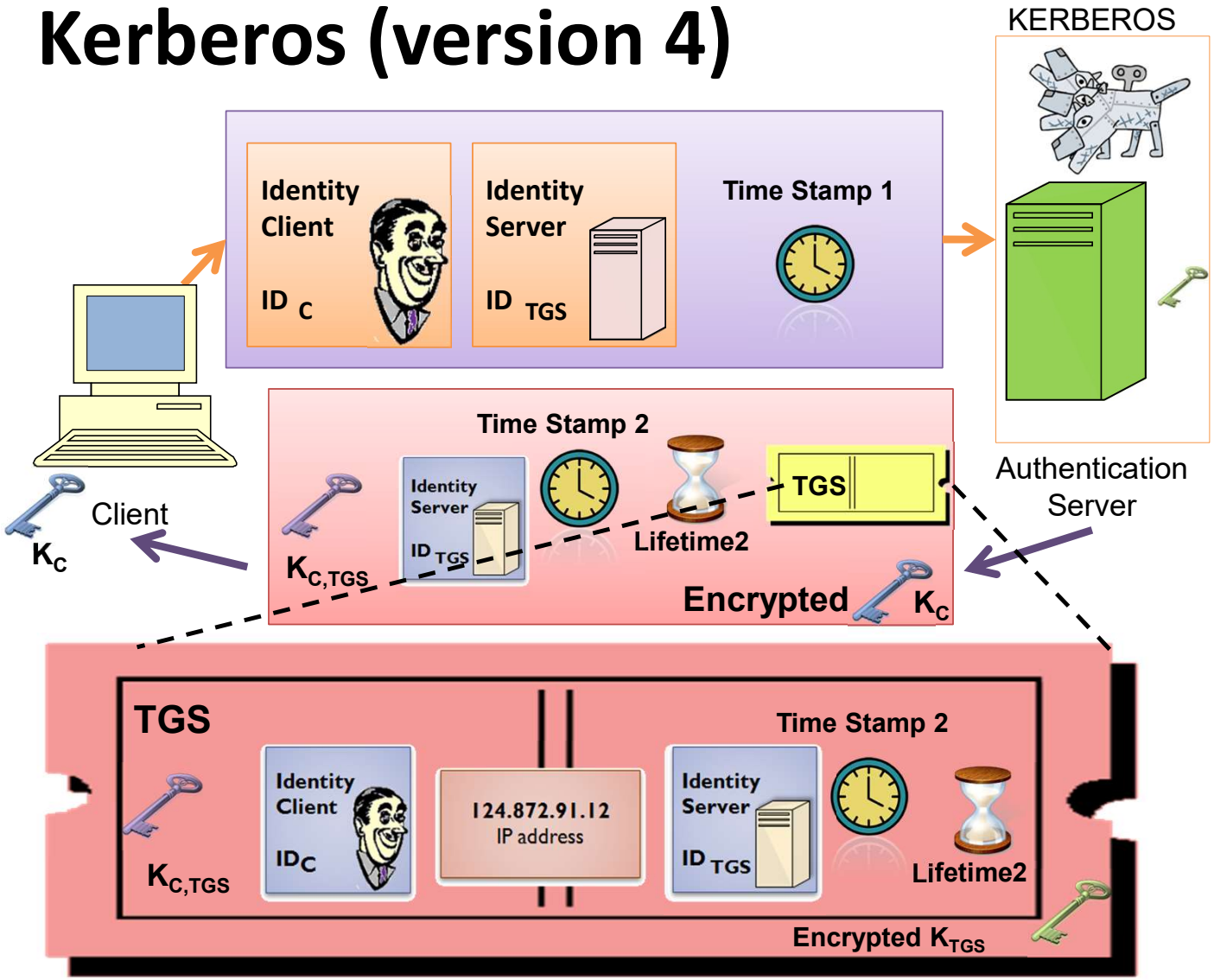
(5) $C \rightarrow V: \quad ID_C \| Ticket_v$

$$Ticket_{tgs} = E(K_{tgs}, [ID_C \| AD_C \| ID_{tgs} \| TS_1 \| Lifetime_1])$$
$$Ticket_v = E(K_v, [ID_C \| AD_C \| ID_v \| TS_2 \| Lifetime_2])$$

Problems

- Lifetime of the ticket
  - Short lifetimes means the user has to type password often.
  - Long lifetimes means that the user has greater opportunity to replay.
  - A network service (the TGS or an application service) must be able to prove that the person using a ticket is the same person to whom that the ticket was issued.
- Server does not authenticate to users. (Why is this a problem?)

Kerberos (version 4)



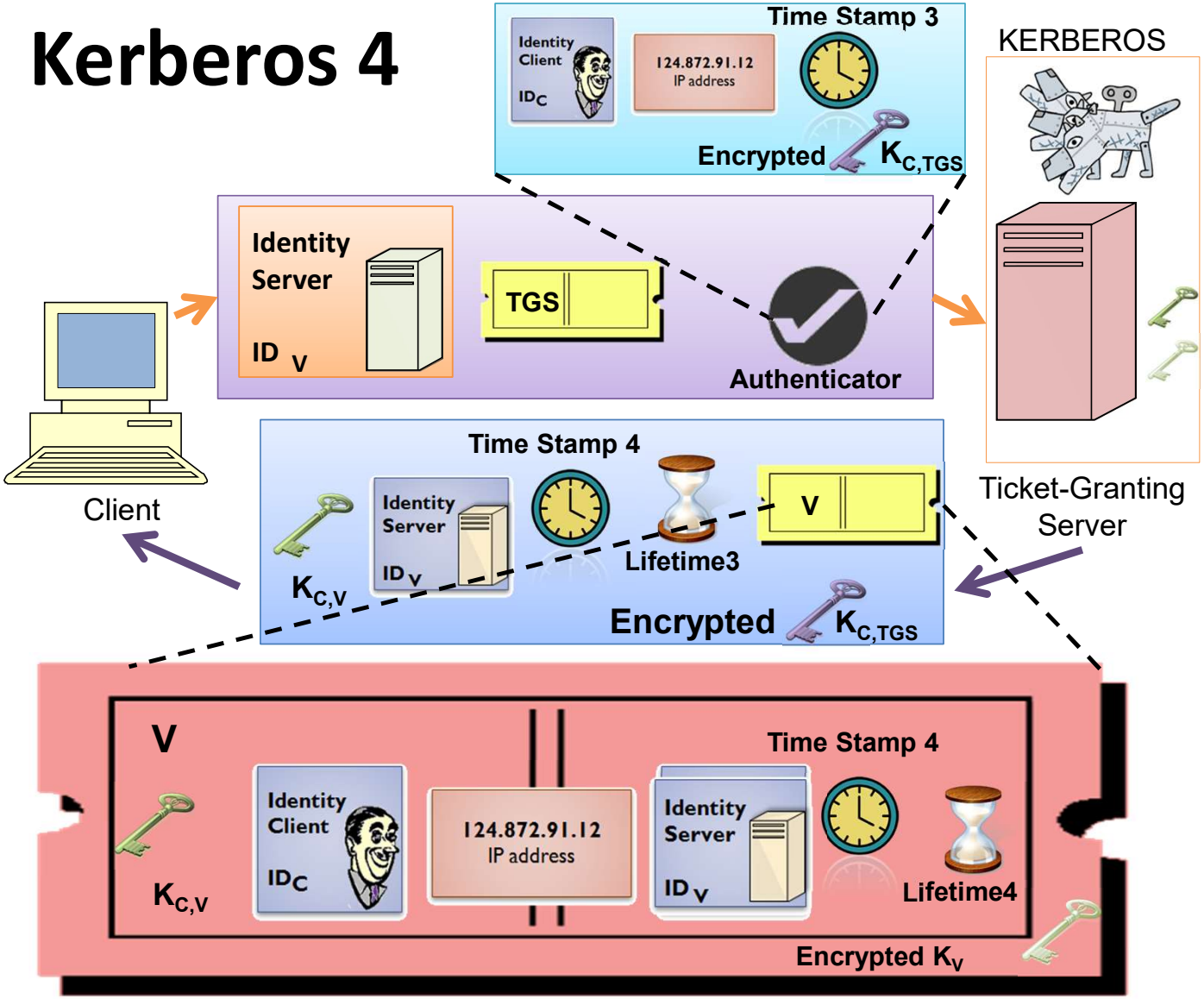
Dialogue

- The client requests access to TGS. It sends the user's ID, the TGS ID and a time-stamp.
- AS responds with an encrypted message. The encryption key is derived from the user's password. The message contains a copy of the session key $K_{C,TGS}$ (the key is generated by AS and used by the client and TGS) and a ticket. The ticket contains a copy of this session key.
- The ticket is encrypted. The key for the ticket (d) encryption is known to AS and TGS (not the client).

Dialogue

- The session key shared by the client and TGS is used by the client to prove its' identity to TGS.
- The time stamp is to verify that the client's clock is synchronised with that of AS.

Kerberos 4


Time Stamp 3
Identity Client
ID C
124.872.91.12
IP address
Encrypted
K C,TGS
KERBEROS
Identity Server
ID V
TGS
Authenticator
Client
Time Stamp 4
Identity Server
ID V
Lifetime3
V
K C,V
Encrypted
K C,TGS
Ticket-Granting Server
V
Time Stamp 4
Identity Client
ID C
124.872.91.12
IP address
Identity Server
ID V
Lifetime4
K C,V
Encrypted K V

Dialogue

- The client request from TGS a service-granting ticket. It sends the user's ID, the desire service ID, the ticket-granting ticket and an authenticator. The authenticator has a very short lifetime and is used to authenticate the user.

- The TGS, decrypts the authenticator using the session key KC,TGS and compares the user's name, address from the authenticator with the ones of the ticket, the ticket is decrypted first. (The authenticator says "at time T_3, I hereby use $K_{C,TGS}$).

Dialogue

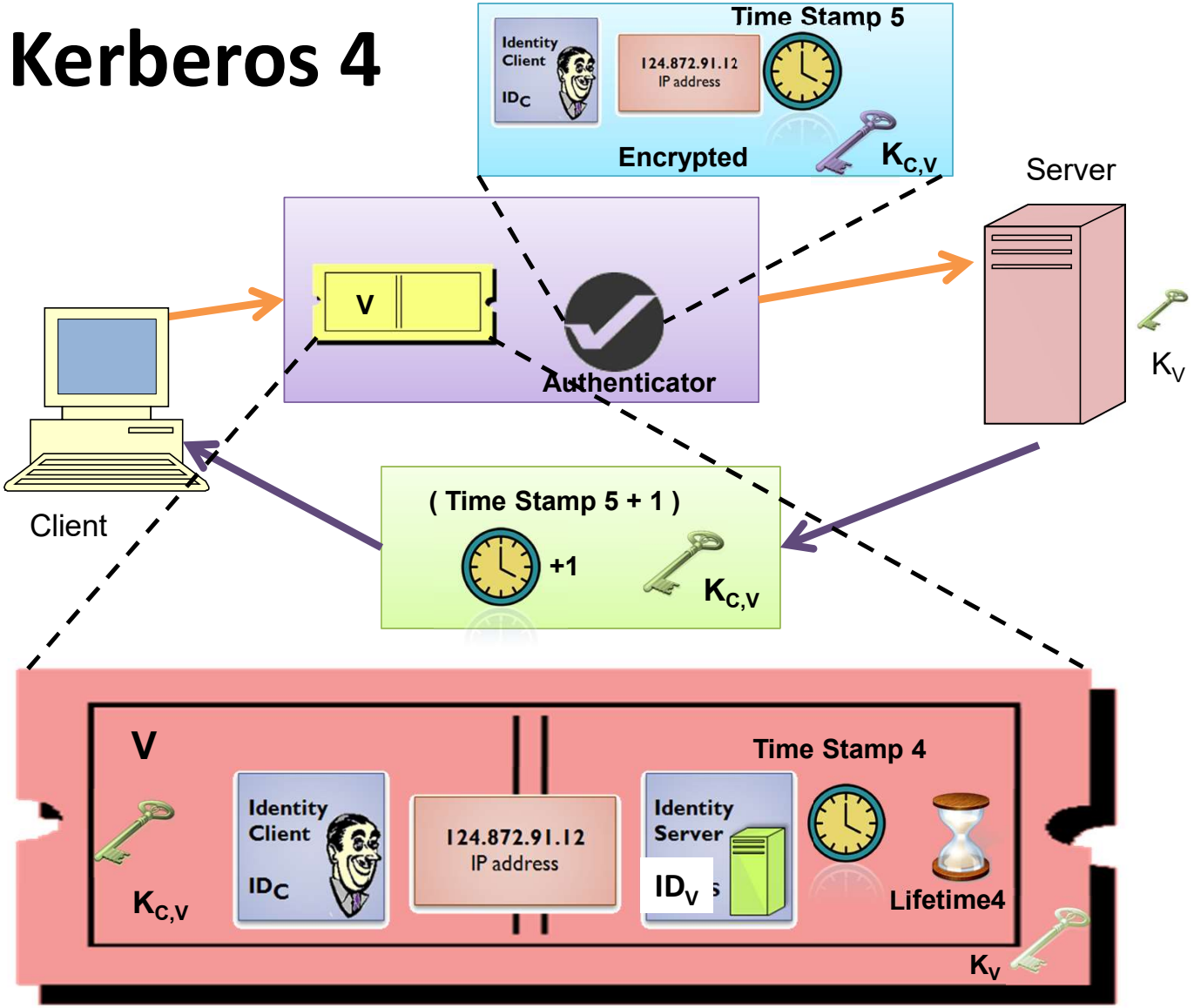
- The reply of TGS includes a new client-server session key (generated by TGS) plus the user's ID, the service-granting ticket and a time-stamp. All these information is encrypted using the session key shared by the TGS and the client.
- The service-granting ticket is encrypted. The key is known to the server and TGS (not the client).
- The ticket-granting ticket assures TGS that this user has been authenticated by AS.
- The authenticator is generated by client to validate the ticket Assures TGS that the ticket presenter is the same as the client whom the ticket was issued.

Dialogue

- The session key shared by client and TGS protects the contents of the message. Is used by TGS to decrypt the authenticator, thereby authenticating the request.

- A copy of the session key accessible to client. Used for secure exchange between the client and server without requiring them to share a permanent key.

Kerberos 4


Time Stamp 5
Identity Client
IDC
124.872.91.12
IP address
Encrypted
KC,V
Server
V
Authenticator
KV
Client
(Time Stamp 5 + 1)
+1
KC,V
V
Time Stamp 4
Identity Client
IDC
124.872.91.12
IP address
Identity Server
IDV
Lifetime4
KC,V
KV

Kerberos

- The client requests access to the server. It sends the service-granting ticket and an authenticator.
- The authenticator contains the user's ID, the address and a time stamp. The authenticator is encrypted with the client-server session key.
- The server authenticates the information by comparing the contents of the ticket and the content of the authenticator.
- The ticket is decrypted using the key that is shared by the server and the TGS.

Kerberos

- If mutual authentication is required, the server sends the time-stamp plus one. This message is encrypted with the session key.

Kerberos v4 structure

- A basic third-party authentication scheme
- Have an Authentication Server (AS)
  - Users initially negotiate with AS to identify self
  - AS provides a non-corruptible authentication credential (ticket granting ticket TGT)
- Have a Ticket Granting server (TGS)
  - Users subsequently request access to other services from TGS on basis of users TGT
- Using a complex protocol using DES

Terms

- C = Client
- AS = authentication server
- V = server
- IDc = identifier of user on C
- IDv = identifier of V
- Pc = password of user on C
- ADc = network address of C
- Kv = secret encryption key shared by the TGS and V
- TS = timestamp
- || = concatenation

Kerberos v4 Dialogue

(1) $\mathbf{C \rightarrow AS}$ $\quad ID_c \,\|\, ID_{tgs} \,\|\, TS_1$

(2) $\mathbf{AS \rightarrow C}$ $\quad E(K_c, [K_{c,tgs} \,\|\, ID_{tgs} \,\|\, TS_2 \,\|\, Lifetime_2 \,\|\, Ticket_{tgs}])$

$$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \,\|\, ID_C \,\|\, AD_C \,\|\, ID_{tgs} \,\|\, TS_2 \,\|\, Lifetime_2])$$

(a) Authentication Service Exchange to obtain ticket-granting ticket

(3) $\mathbf{C \rightarrow TGS}$ $\quad ID_v \,\|\, Ticket_{tgs} \,\|\, Authenticator_c$

(4) $\mathbf{TGS \rightarrow C}$ $\quad E(K_{c,tgs}, [K_{c,v} \,\|\, ID_v \,\|\, TS_4 \,\|\, Ticket_v])$

$$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \,\|\, ID_C \,\|\, AD_C \,\|\, ID_{tgs} \,\|\, TS_2 \,\|\, Lifetime_2])$$

$$Ticket_v = E(K_v, [K_{c,v} \,\|\, ID_C \,\|\, AD_C \,\|\, ID_v \,\|\, TS_4 \,\|\, Lifetime_4])$$

$$Authenticator_c = E(K_{c,tgs}, [ID_C \,\|\, AD_C \,\|\, TS_3])$$

(b) Ticket-Granting Service Exchange to obtain service-granting ticket

(5) $\mathbf{C \rightarrow V}$ $\quad Ticket_v \,\|\, Authenticator_c$

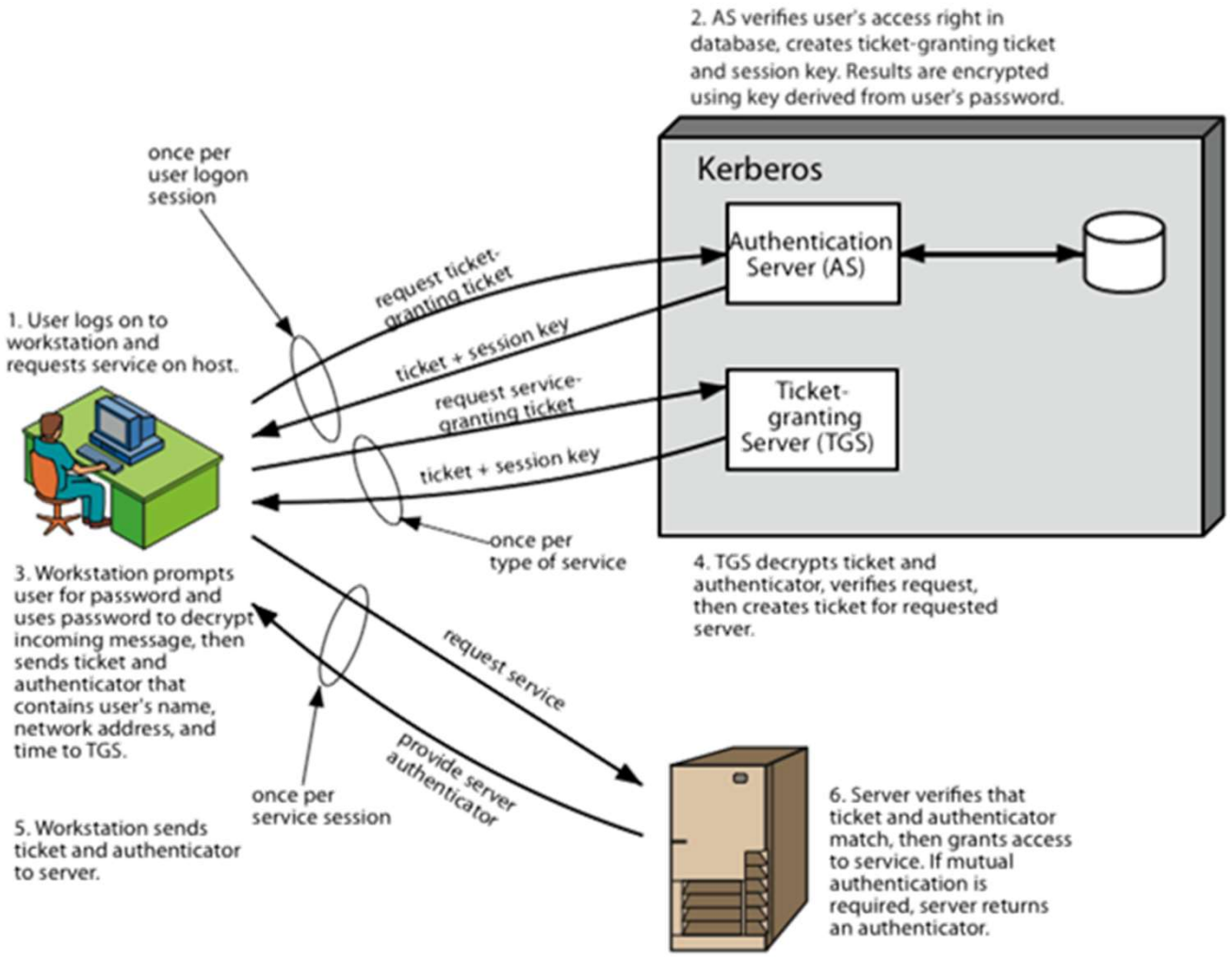
(6) $\mathbf{V \rightarrow C}$ $\quad E(K_{c,v}, [TS_5 + 1])$ (for mutual authentication)

$$Ticket_v = E(K_v, [K_{c,v} \,\|\, ID_C \,\|\, AD_C \,\|\, ID_v \,\|\, TS_4 \,\|\, Lifetime_4])$$

$$Authenticator_c = E(K_{c,v}, [ID_C \,\|\, AD_C \,\|\, TS_5])$$

(c) Client/Server Authentication Exchange to obtain service

Kerberos v4 Summary



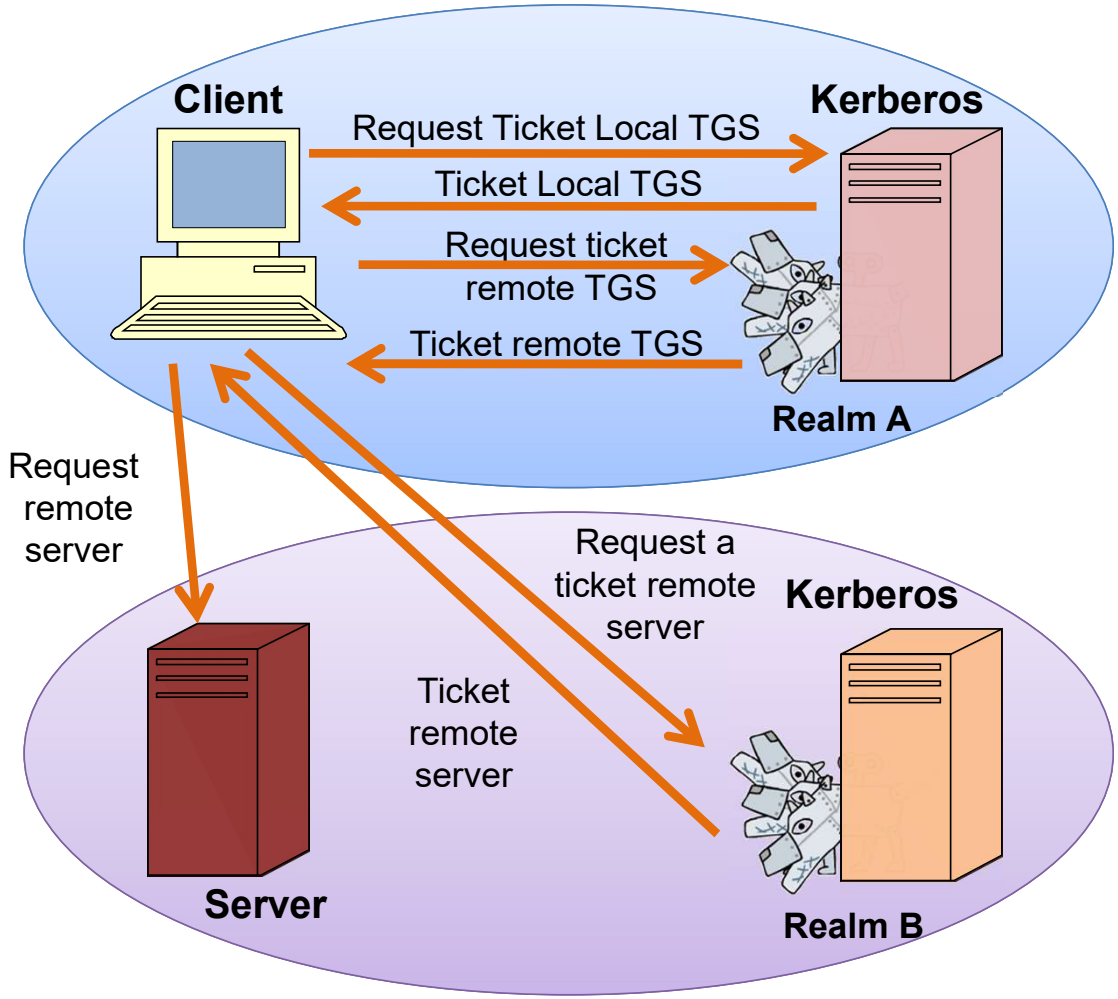
Kerberos Environment

- Consist of a Kerberos server, a number of clients, and a number of application servers. This environment is called **realm**. With the following:
  - The Kerberos server must have the ID and password of all users. All users are registered with the Kerberos server.
  - The Kerberos server must share secret keys with each server. All servers are registered with the Kerberos server.

Kerberos realms

– For inter-realm authentication the Kerberos server in each realm shares a secret key with the server in the other realm. The two Kerberos server are registered with each other.

Kerberos realms



Kerberos version 5

- Developed in mid 1990's
- Specified as Internet standard RFC 1510

- Provides the following improvements over version 4

  – Environmental
    - Any encryption technique can be used (not only DES)
    - Different network address type can be used (not only IP)
    - Unambiguous byte ordering
    - Longer ticket lifetimes
    - Authentication forwarding
    - Inter-realm authentication.

Kerberos version 5

- **Technical**
  - Avoid double encryption
  - CBC mode of operation instead of PCBC.
  - Sub-session keys for client and server (better security)
  - Improvements against password attacks.

Kerberos v5 Dialogue

(1) C → AS Options ‖ ID_c ‖ $Realm_c$ ‖ ID_{tgs} ‖ $Times$ ‖ $Nonce_1$

(2) AS → C $Realm_c$ ‖ ID_C ‖ $Ticket_{tgs}$ ‖ $E(K_c, [K_{c,tgs} \| Times \| Nonce_1 \| Realm_{tgs} \| ID_{tgs}])$

$Ticket_{tgs} = E(K_{tgs}, [Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times])$

(a) Authentication Service Exchange to obtain ticket-granting ticket

(3) C → TGS Options ‖ ID_v ‖ $Times$ ‖ ‖ $Nonce_2$ ‖ $Ticket_{tgs}$ ‖ $Authenticator_c$

(4) TGS → C $Realm_c$ ‖ ID_C ‖ $Ticket_v$ ‖ $E(K_{c,tgs}, [K_{c,v} \| Times \| Nonce_2 \| Realm_v \| ID_v])$

$Ticket_{tgs} = E(K_{tgs}, [Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times])$

$Ticket_v = E(K_v, [Flags \| K_{c,v} \| Realm_c \| ID_C \| AD_C \| Times])$

$Authenticator_c = E(K_{c,tgs}, [ID_C \| Realm_c \| TS_1])$

(b) Ticket-Granting Service Exchange to obtain service-granting ticket

(5) C → V Options ‖ $Ticket_v$ ‖ $Authenticator_c$

(6) V → C $E_{K_{c,v}}$ [TS_2 ‖ Subkey ‖ Seq#]

$Ticket_v = E(K_v, [Flags \| K_{c,v} \| Realm_c \| ID_C \| AD_C \| Times])$

$Authenticator_c = E(K_{c,v}, [ID_C \| Realm_c \| TS_2 \| Subkey \| Seq\#])$

(c) Client/Server Authentication Exchange to obtain service

Summary

- Kerberos
  - **Definition:** Kerberos is an authentication service designed for use in a distributed environment.
  - **Uses:** It makes use of a trusted third-party authentication service that enables clients and servers to establish authenticated communication.

Security at the IP layer

IP Security

IP Security

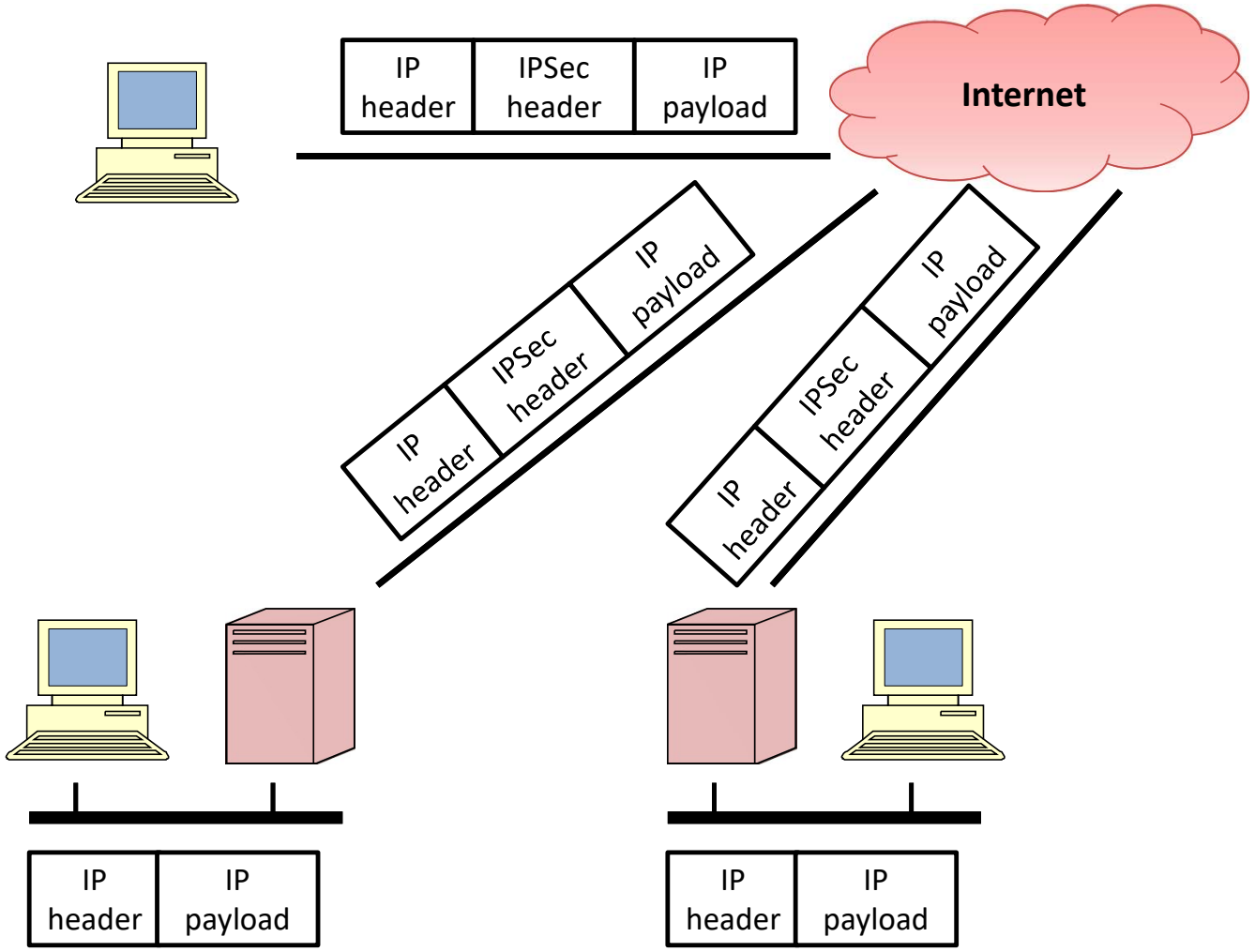
- There are many application-specific security mechanism in a number of application areas, e.g. Electronic mail (S/MIME,PGP), client/server (Kerberos), Web access (Secure Socket Layer) etc.

- Security at the IP level (IPSec) can ensure security not only for applications that have security mechanisms but for many security-ignorant application.

IPSec

- IP security is built into the IP layer.
- IPSec can encrypt and/or authenticate *all* traffic at the IP authentication. Comprised of two pairs:
  - IPSEC proper (authentication and encryption)
  - IPSEC key management
- Required for IPv6, optional for IPv4.

IPSec Typical Configuration



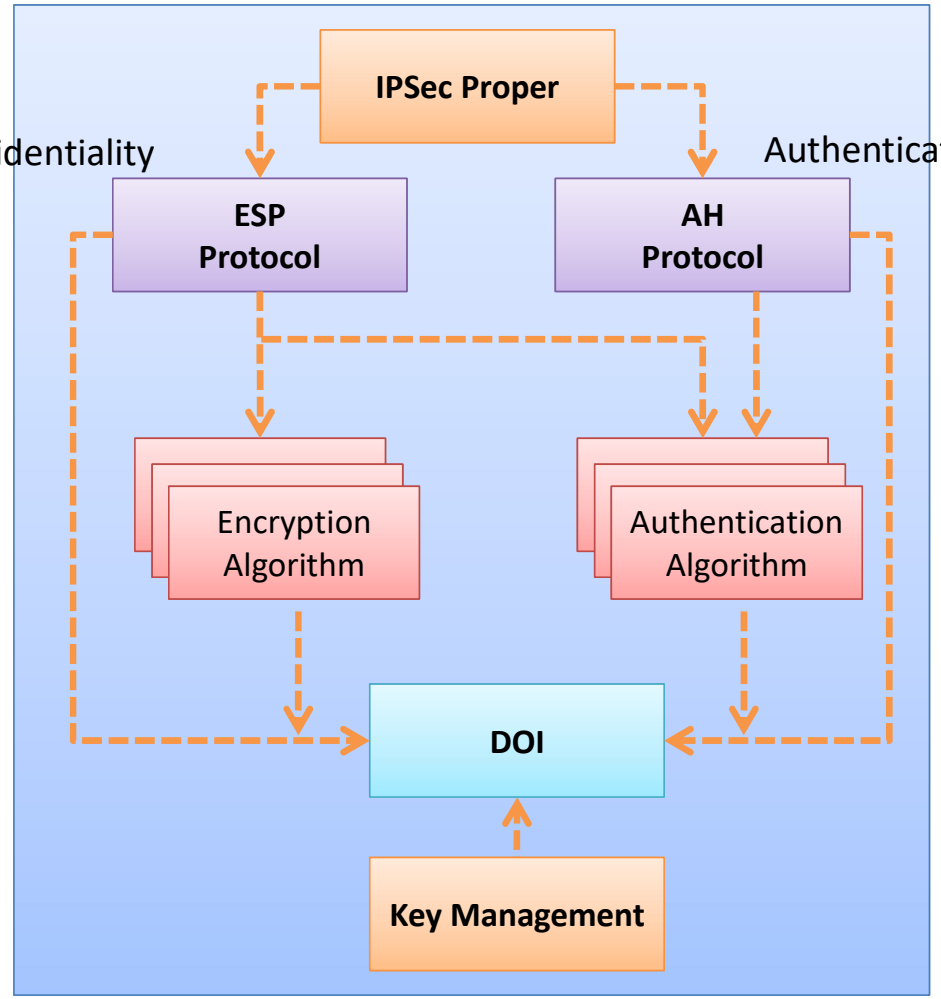
IPSec Benefits

- Combined with a Firewall or router it provides strong security to all traffic crossing the perimeter.
- IPSec in a Firewall is resistant to bypass if all traffic from the outside must use IP and the Firewall is the only gate from the Internet.
- IPSec is below the transport layer (TCP, UDP), hence is transparent to applications.
- IPSec is transparent to end users.

IPSec Typical Configuration

- An organisation keeps LANs at dispersal locations.
- The IPSec protocols operate in networking devices (routers, Firewalls).
- Applications
  - Secure LAN and WAN connectivity i.e. a VPN in college.
  - Secure remote access.
  - Secure communication with other organisations.
  - E-commerce security.

IPSec Architecture



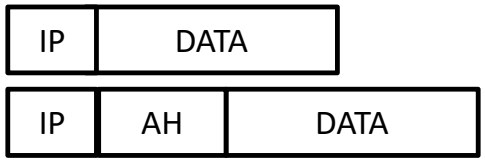
- Encapsulate Security Payload (ESP)
- Authentication Header (AH)
- Domain of Interpretation (DOI)

IPSec Services

	AH	ESP	
	AH	ESP encryption	ESP encryption with authentication
Access control	✓	✓	✓
Connectionless integrity	✓		✓
Data origin authentication	✓		✓
Rejection of replayed packets	✓	✓	✓
Confidentiality (encryption)		✓	✓
Limited traffic flow confidentiality		✓	✓

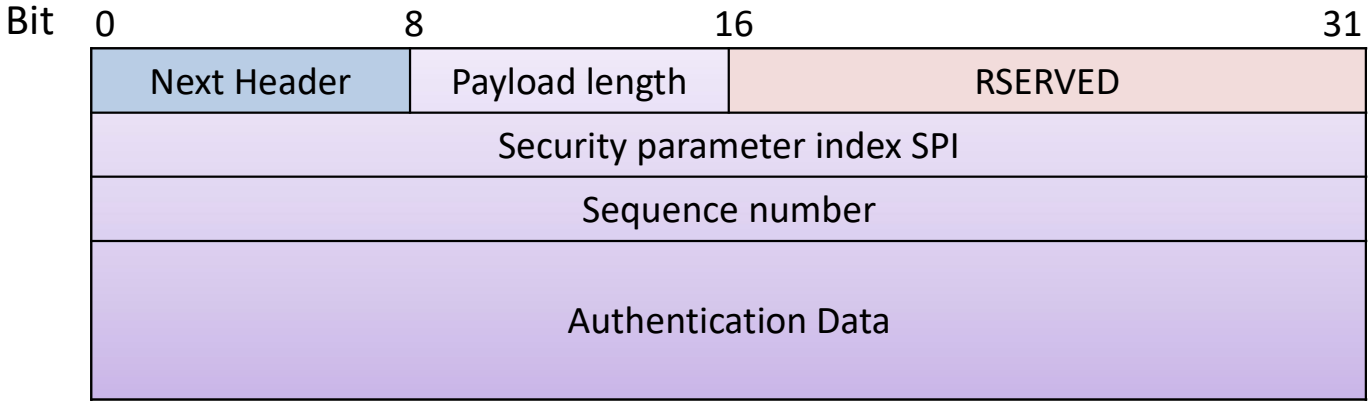
Authentication Header (AH)

- Integrity protection only.
- Inserted in the datagram



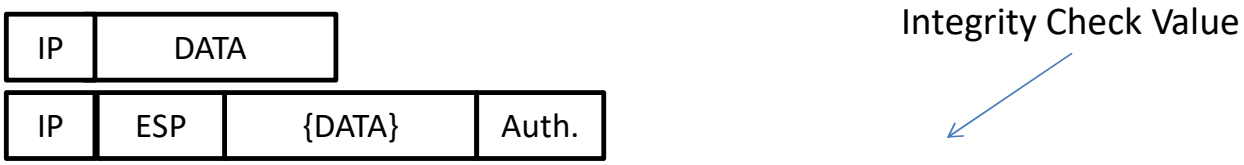
  - Integrity check value (ICV) is 96-bit HMAC.
  - Authenticates entire datagram;
    - Mutable fields (time-to-live, IP checksums) are ignored before the AH is added.

Authentication Header



ESP

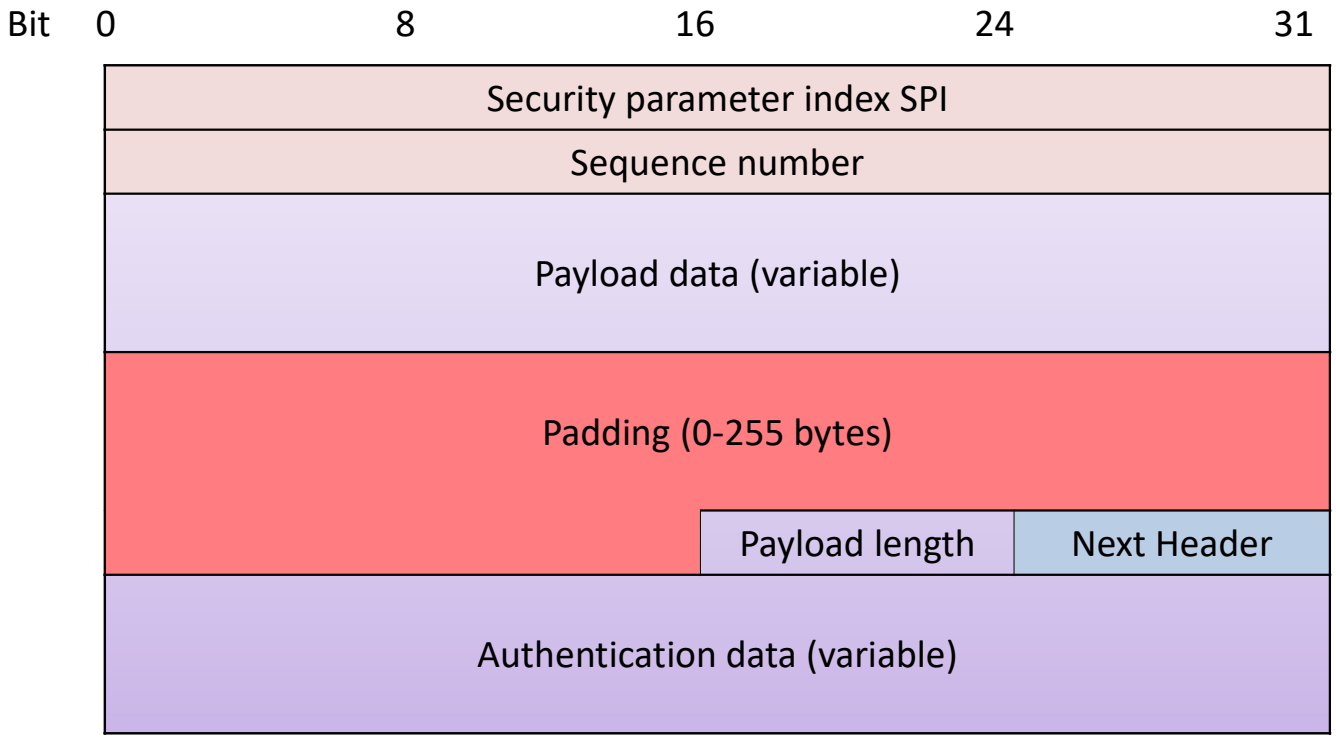
- Provides authentication (optional) and confidentiality
- Inserted in the datagram



Contains sequence numbers and optional ICV as for AH.

- Encryption protects payload
- Authentication protects header and encryption

Encapsulate Security Payload (ESP)



IPSec Algorithms

- Encryption: DES in CBC mode
- Authentication: HMAC/MD5 and HMAC/SHA (truncated to 96 bits)

Later versions added optional DOI-dependent algorithms

- TDES
- Blowfish
- CAST-128
- IDEA
- RC5

Domain of Interpretation (DOI)

- From the Internet IP Security Domain of Interpretation for ISAKMP, D. Diper November 1998, http://www.ietf.org/rfc/rfc2407.txt

- *The Internet Security Association and Key Management Protocol (ISAKMP) defines a framework for security association management and cryptographic key establishment for the Internet. This framework consists of defined exchanges, payloads, and processing guidelines that occur within a given Domain of Interpretation (DOI). This document defines the Internet IP Security DOI (IPSEC DOI), which instantiates ISAKMP for use with IP when IP uses ISAKMP to negotiate security associations.*

Domain of Interpretation (DOI)

- Contains values to relate the different specifications of the protocol
  - Identifiers for encryption and authentication algorithms
  - Operational parameters, key lifetimes, key exchange etc.

Security Association (SA)

- A one-way relationship between sender and receiver that describes a security service.
- For two-way exchange of data, two SAs are needed, from sender-to-receiver and receiver-to-sender.

SAs are defined by three parameters	
Security Parameters Index (SPI)	A label (bit string) to identify the security association
IP Destination Address	Only unicast
Security Protocol Identifier	AH or ESP

Parameters which characterise the nature of a given SA

- Sequence Number Counter
- Sequence Counter Overflow
- Anti–Replay Window
- AH Information
- ESP Information
- Lifetime of this Security Association
- IPSec Protocol Mode: tunnel or transport
- Path MTU (maximum transmission unit)

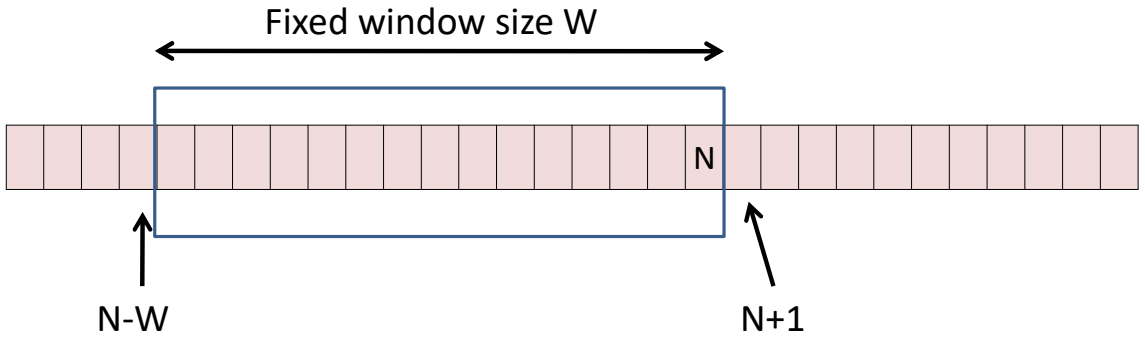
e.g. Super packets

Anti-Replay Service

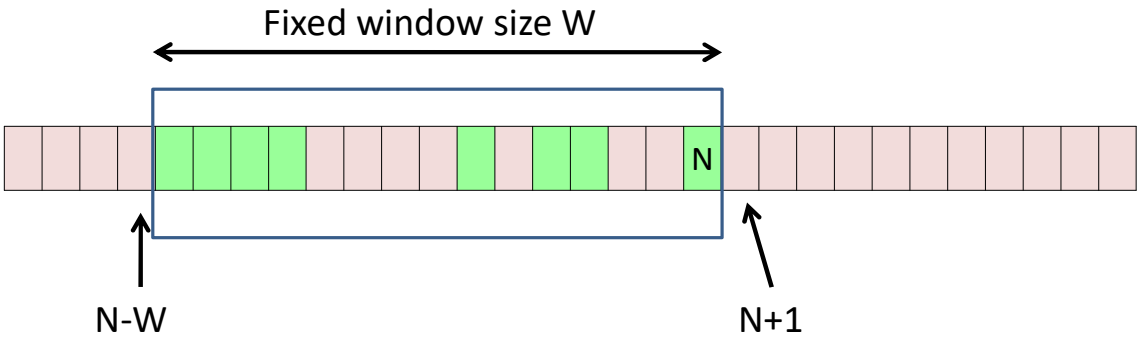
- To disrupt in some way, the attacker obtains a copy of an authenticated packet and re–transmit the packet to its intended destination.
  - Sequence Number used to stop this attack
  - Packets are numbered (using a counter) as they are sent
  - If sequence number greater than $2^{64} - 1$ then terminate SA and start again
  - As IP is connectionless and unreliable, use a sliding window to overcome these IP limitations.

Anti-Replay Service

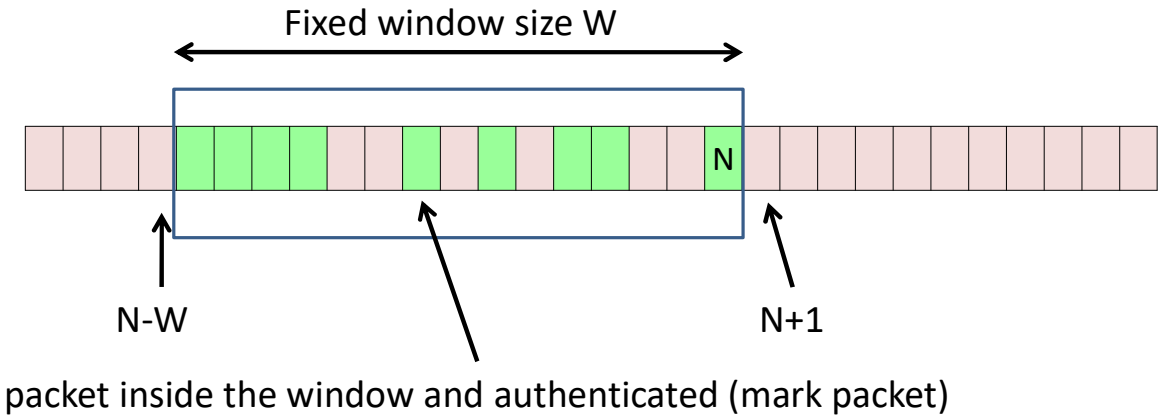
- Sender implements a window of size W.
- Right edge of the window represents highest sequence number *N*.
- Left edge of the window is *N* – W



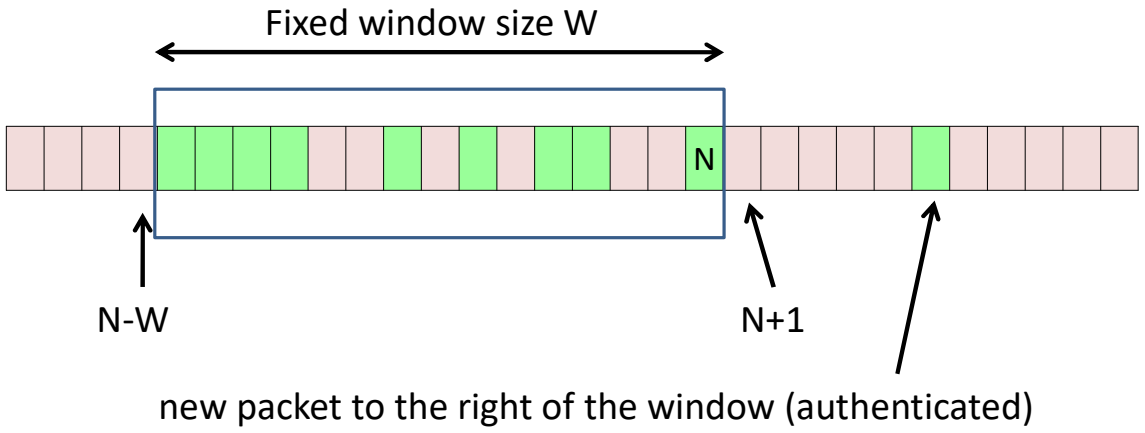
Anti-Replay Service



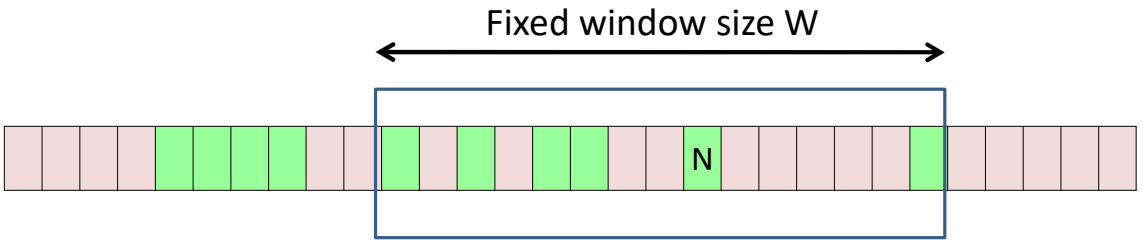
Anti-Replay Service



Anti-Replay Service

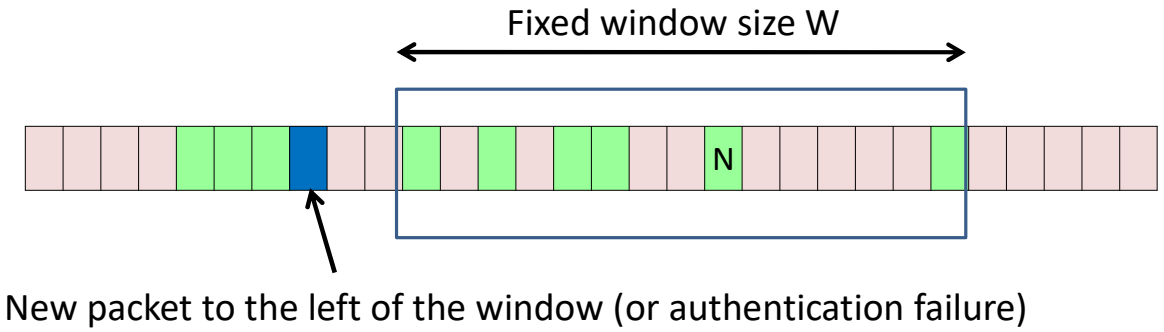


Anti-Replay Service

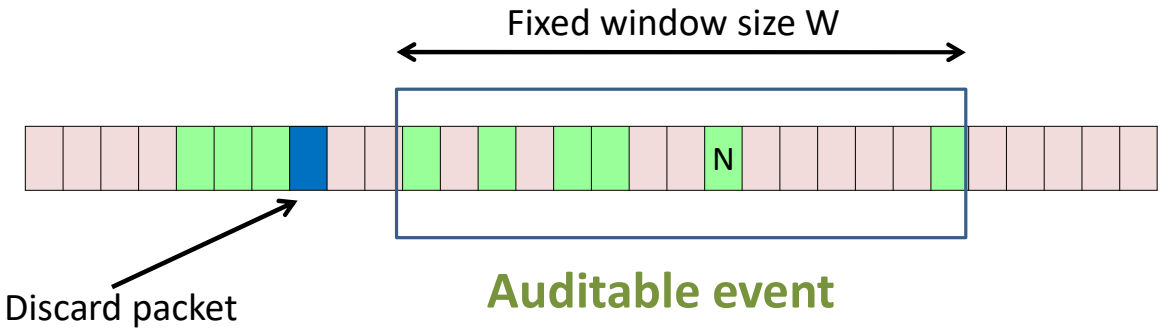


Slide the window so that this new packet is its right edge

Anti-Replay Service



Anti-Replay Service



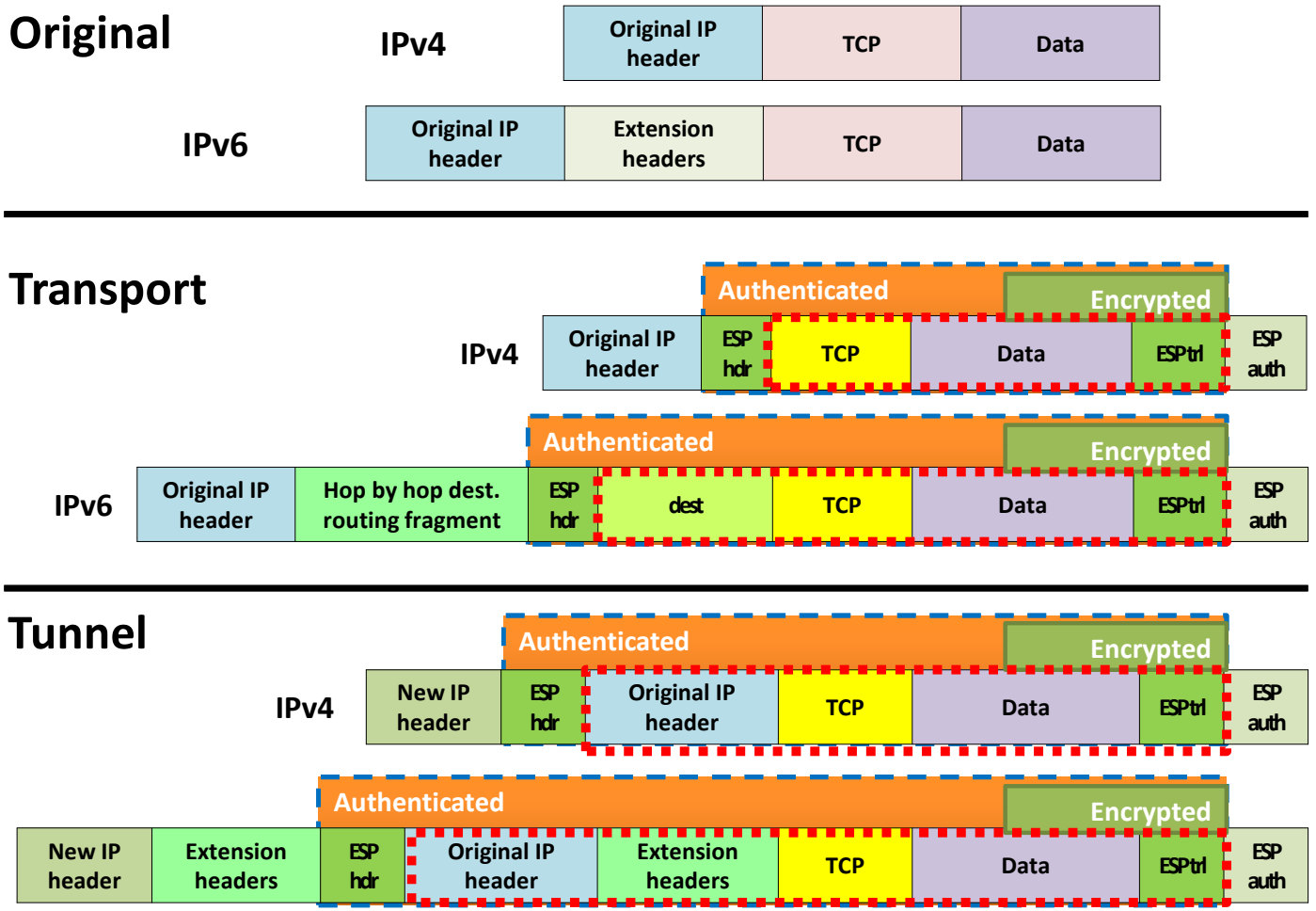
Processing

- Use Security Parameter Index (SPI) to look up security association (SA).
- Perform authentication check using SA
- Perform encryption/decryption of authenticated data using SA

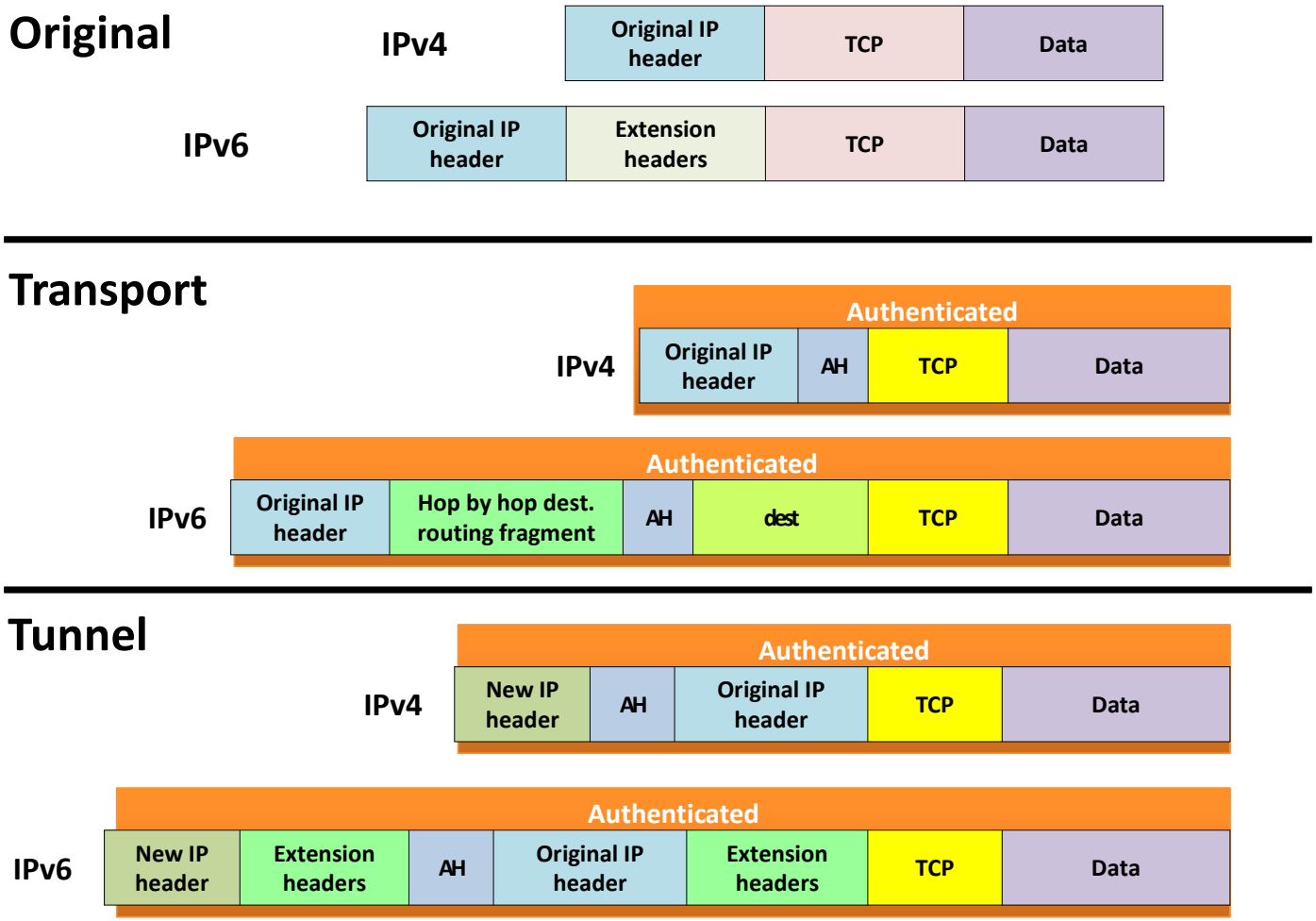
Operates in two modes:

- Tunnel Mode: protects entire packet.
- Transport Mode: protects payload.

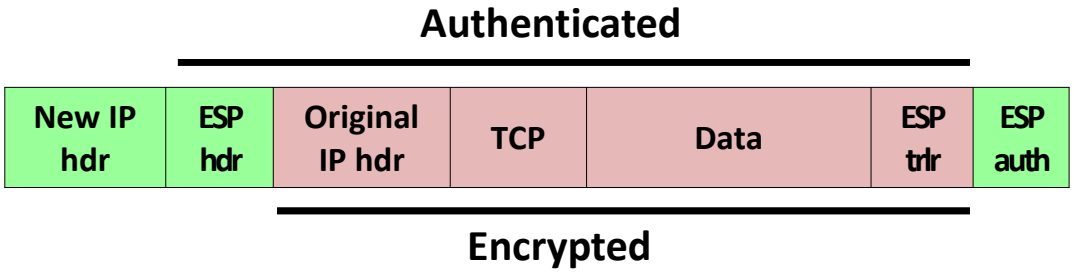
Tunnel and Transport mode (ESP)



Tunnel and Transport mode (AH)

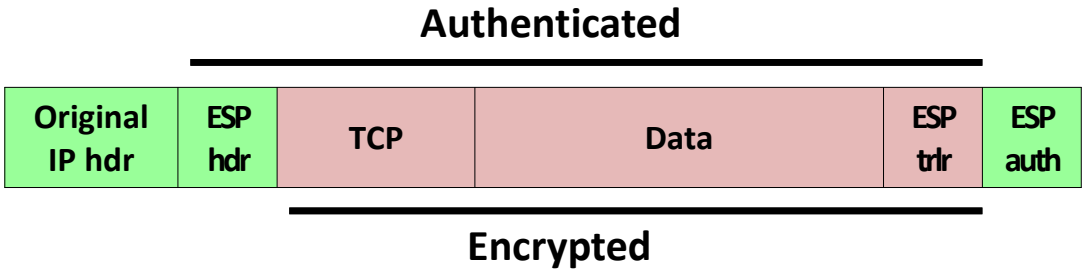


ESP Tunnel Mode



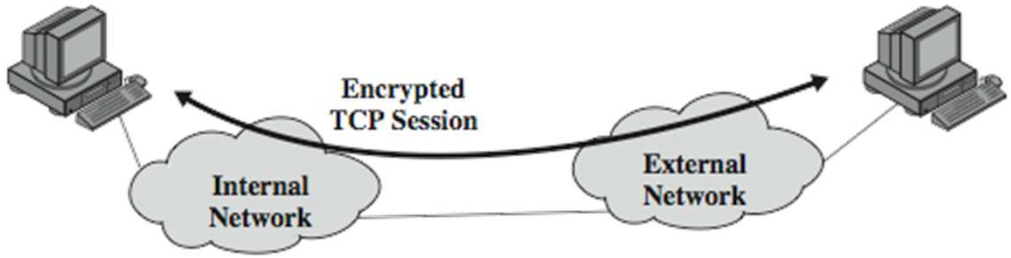
- In tunnel mode ESP can be used to set up a virtual private network.
- Host on the internet networks use the Internet transport of data but do not interact with other Internet-based hosts.
- Tunnel mode operation provides protection against traffic analysis.

ESP Transport Mode

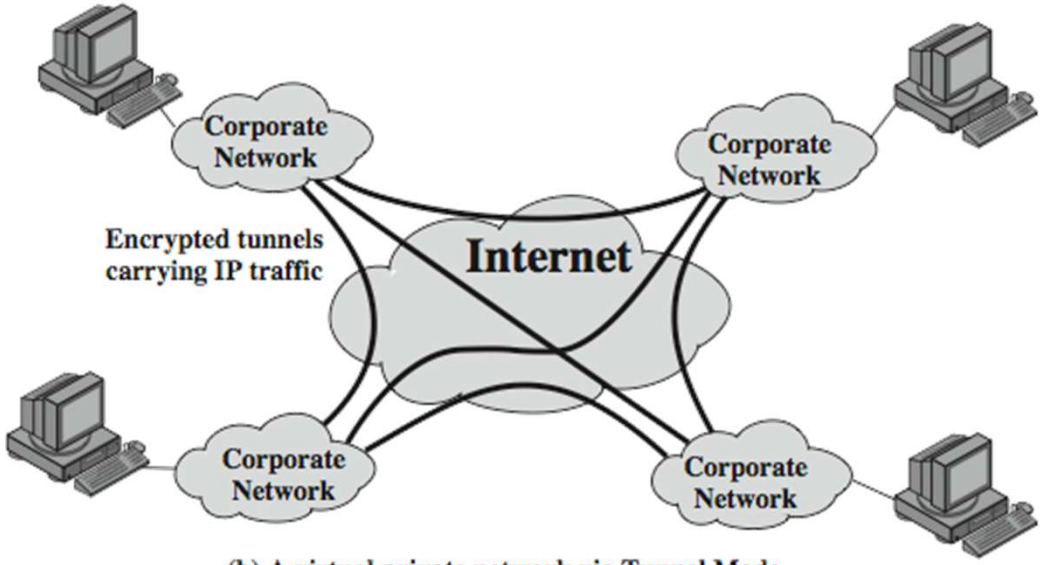


- In transport mode ESP is used to encrypt (and optional to authenticate) the data carried by IP.
- The entire transport-level segment plus the ESP trailer are encrypted.
- Transport mode operation provides confidentiality.

Transport and Tunnel Modes



(a) Transport-level security



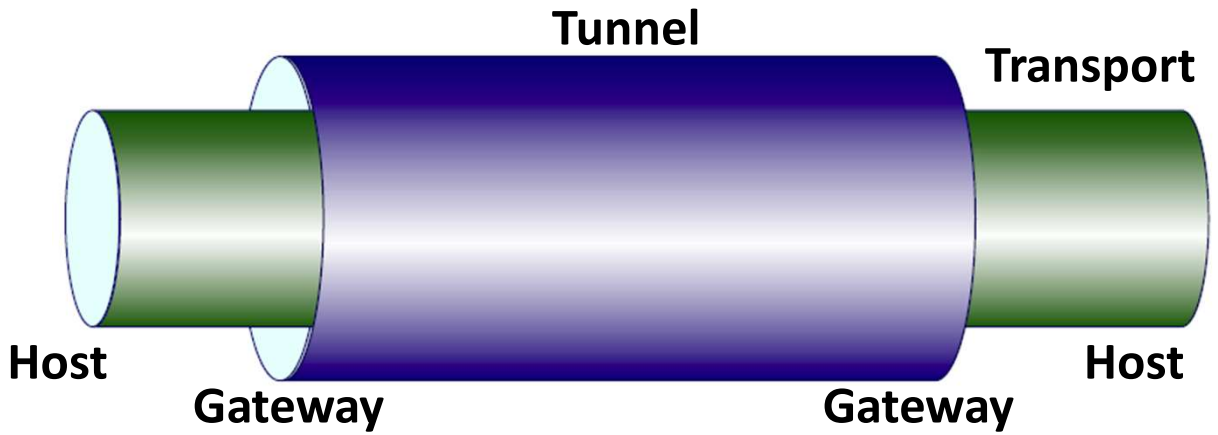
(b) A virtual private network via Tunnel Mode

Combining Security Associations

- SA's can implement either AH or ESP
- To implement both need to combine SA's
  - Form a security association bundle
  - May terminate at different or same endpoints
  - Combined by
    - Transport adjacency
    - Iterated tunneling
- Combining authentication & encryption
  - ESP with authentication, bundled inner ESP & outer AH, bundled inner transport & outer ESP

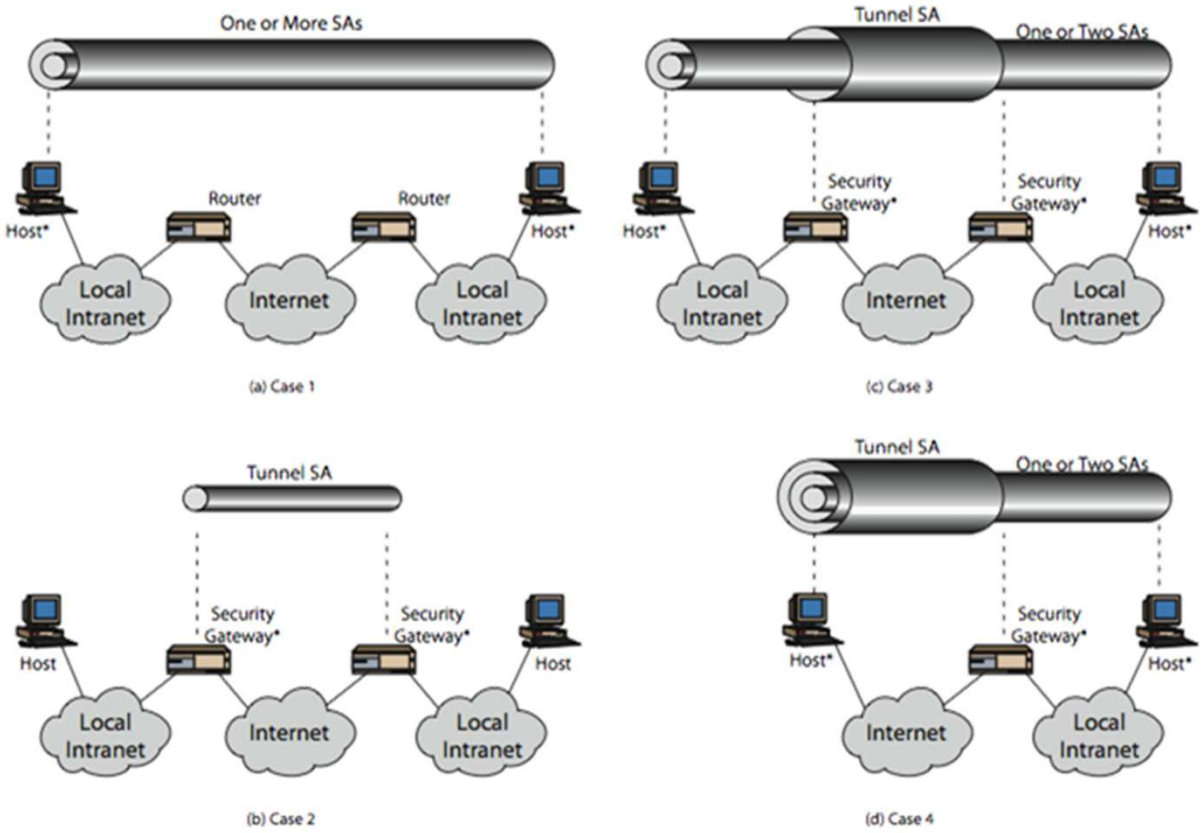
Combination of Security Association

- Example
  - Authentication between host (without encryption).
- AH-transport between hosts
  - Confidentiality when transversing the WAN. ESP-tunnel between gateways.



Combining Security Associations

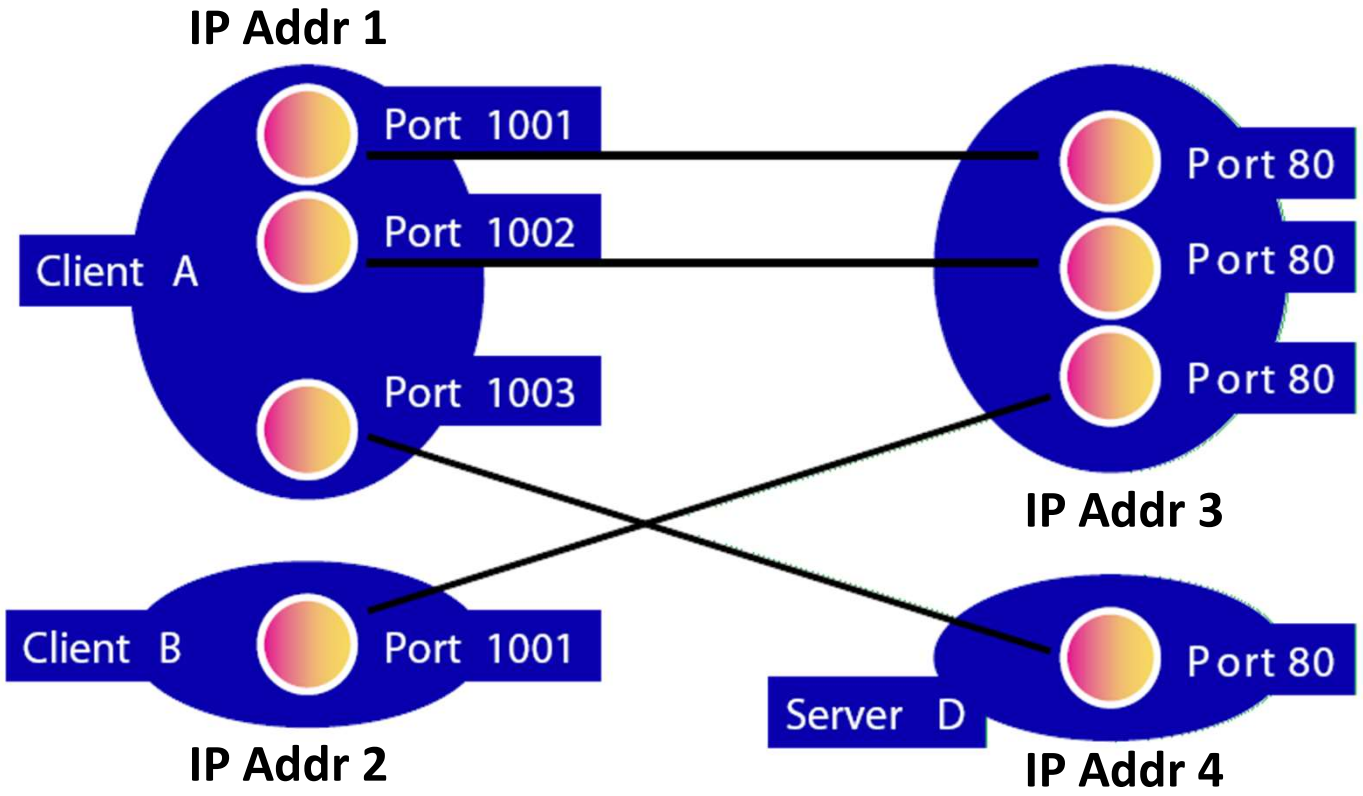
More Examples



Sockets

- A Socket is a communication end point
- Applications communication using a socket
- Application Program Interface (API)

Sockets


IP Addr 1
Port 1001
Port 1002
Port 1003
Client A
Port 80
Port 80
Port 80
IP Addr 3
Client B
Port 1001
IP Addr 2
Server D
Port 80
IP Addr 4

Sockets

- A socket is defined in the operating system as a structure.
  - **Family:** defines the protocol group: IPv4, IPv6...
  - **Type:** defines the type i.e. stream, datagram, or raw socket
  - **Protocol:** usually set to zero for TCP and UDP
  - **Local socket address:** defines the local socket address, a structure of type sockaddress
  - **Remote socket address:** defines the remote socket address, a structure of type sockaddress

IPSec: Packet or Socket

IPsec policy can be configured in per-packet, or per-socket manner:

- **Per-packet:** configured into the kernel just like packet filters. You can specify like ‘‘encrypt outgoing packets if I’m sending to 10.1.1.0/24’’. This works well when you are running an IPsec router.

- **Per-socket:** configured via setsockopt(2) for a certain socket. You can specify like ‘‘encrypt outgoing packets from this socket’’. This works well when you would like to run IPsec-aware server program.

Key Management

- Key management involves the determination and distribution of secret keys
- The distribution of keys can be manual or automatic.
- Oakley Key Determination Protocol: is a key exchange protocol (improved *Diffie–Hellman*)
- Internet Security Association and Key Management Protocol (**ISAKMP**) provides a framework for key management and provides the specific protocol support.

Oakley

- Is a key exchange protocol, based on *Diffie-Hellman* key exchange

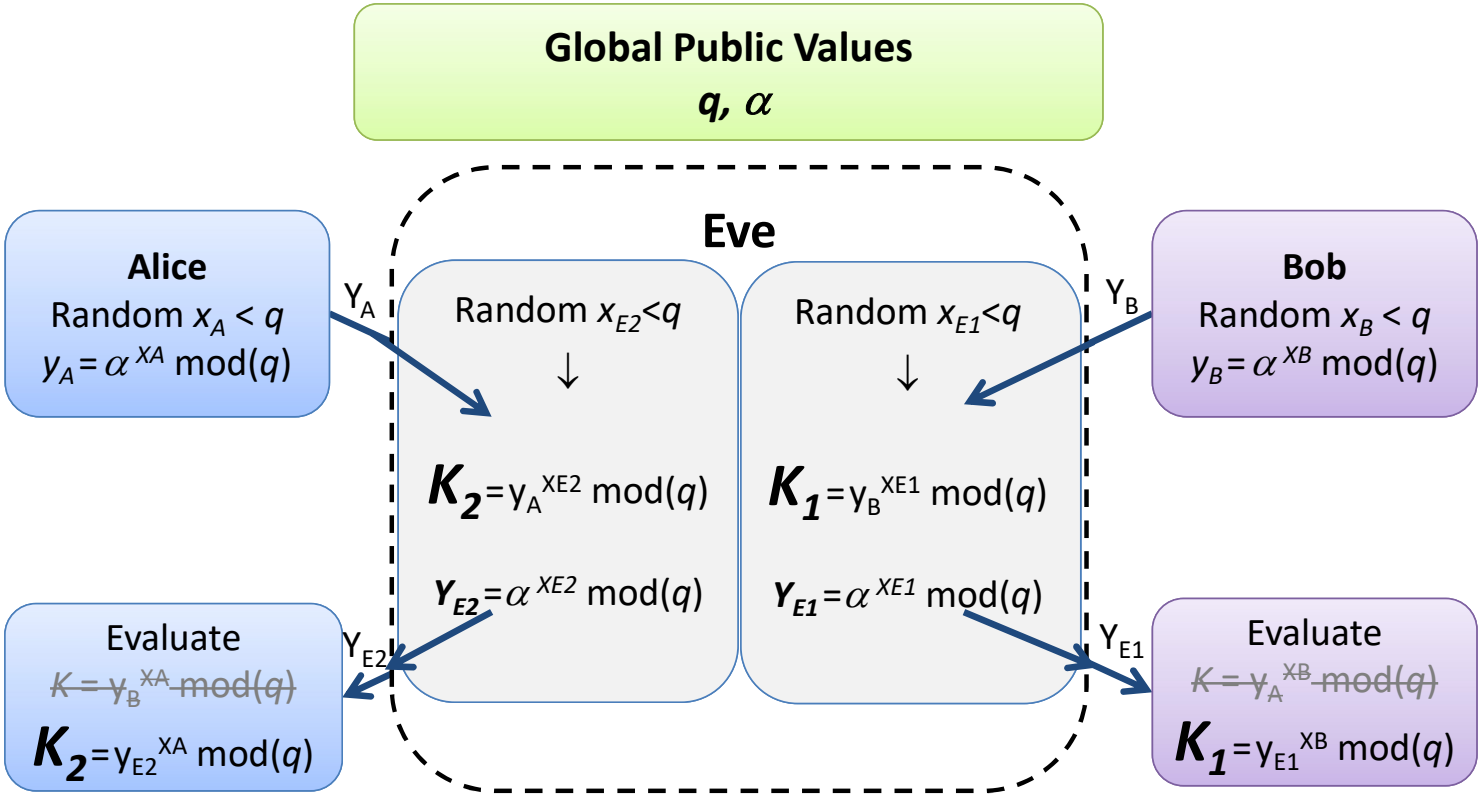
- Exchange messages containing any of
  - Client/Server cookies
  - *Diffie-Hellman* information
  - Offered/chosen security parameters
  - Client/Server ID's

until both sides are satisfied.

Oakley is very open–ended, with many variations possible, ex

  - Speed vs thoroughness
  - New session vs re-key
  - Identification vs anonymity
  - *Diffie–Hellman* vs shared secrets vs Public Key -based exchange

Man-in-the-middle attack



Oakley Authentication

- Digital Signature: Authentication is done by signing a mutually obtainable hash. Each party encrypts the hash with its private key.
- Public–key encryption
- Symmetric–key encryption

Oakley Characteristics

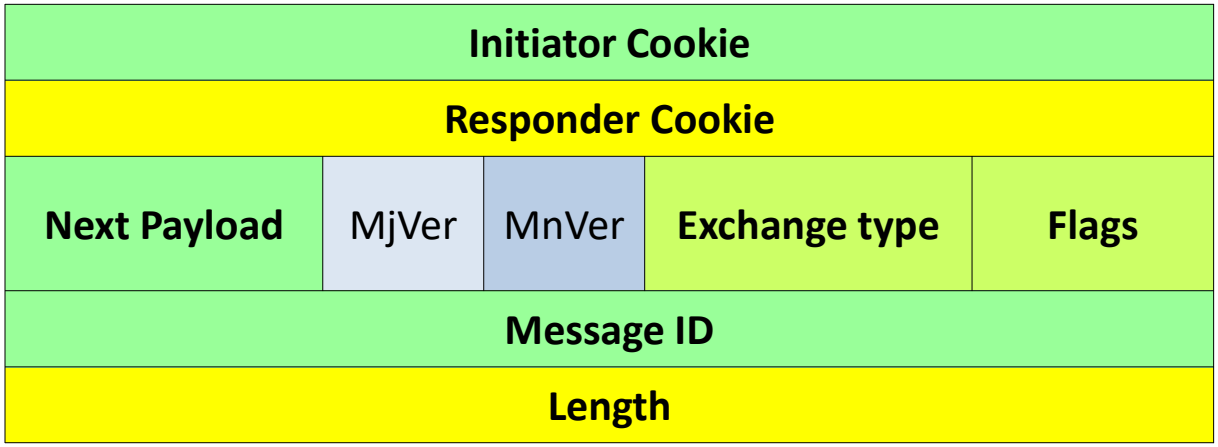
- Use of cookies to thwart clogging attacks
- Specify the global parameters of the *Diffie–Hellman* algorithm
- Uses nonce to ensure against replay attacks
- Exchange of *Diffie–Hellman* public key values
- Authenticates the *Diffie–Hellman* exchange to thwart man-in-the-middle attack

*Clogging attack is a denial of service attack.

ISAKMP (Internet Security Association and Key Management Protocol)

- NSA–designed protocol to exchange security parameters (but not establish key)
  - Protocol to establish, modify and delete IPSec security association
  - General framework for exchanging cookies, security parameters, key management and identification
  - Details left to other protocols
- Two phases:
  1. Establish secure, authenticate channel (SA)
  2. Negotiate security parameters (KMP)

ISAKMP



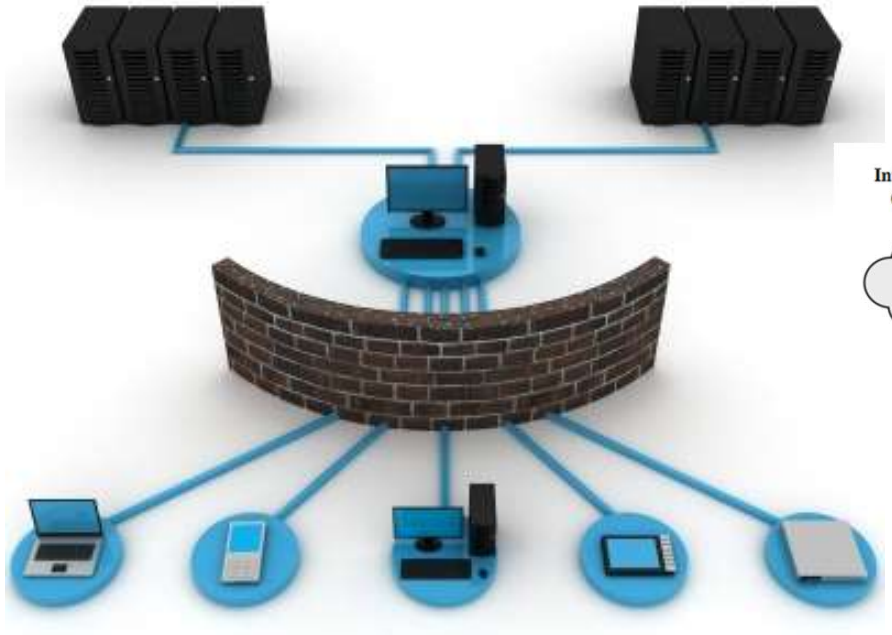
ISAKMP header

Generic payload header

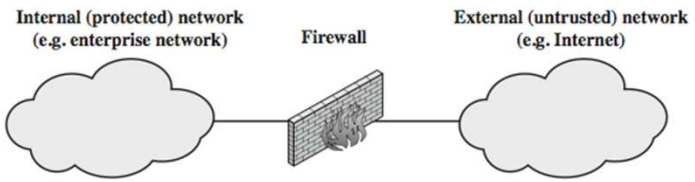
ISAKMP / Oakley

- They merged so
  - ISAKMP provides the protocol framework
  - Oakley provides the security mechanism

Combined version clarifies both protocols, resolve ambiguities.



Firewalls


Internal (protected) network
(e.g. enterprise network)
Firewall
External (untrusted) network
(e.g. Internet)

Firewalls

- A Firewall protects a local system/network from network–based security threats, at the same time allows access to the outside world (WAN, Internet).

- A Firewall is needed because by all networks
  - Internet connectivity is not an option
  - Internet access creates a security threat to the local network
  - It is not economical to equip each PC, server, etc. with strong security features

Firewalls

- The Firewall is inserted between the premises network and the Internet.
- The Firewall establishes a controlled link and security wall between the premises and the Internet.
- The Firewall may be one or more computer systems.
- All traffic from inside to outside and vice versa must pass through the Firewall.
- Only authorised traffic will be allowed to pass.
- The Firewall itself is immune to penetration.

Firewall Characteristics

- **Firewall also provides**
  - Service Control
  - Direction Control
  - Use Control
  - Behaviour Control

Firewall Limitations

- The Firewall cannot protect against attacks that bypass the Firewall.
- The Firewall does not protect against internal attacks.
- The Firewall cannot protect against computer viruses.

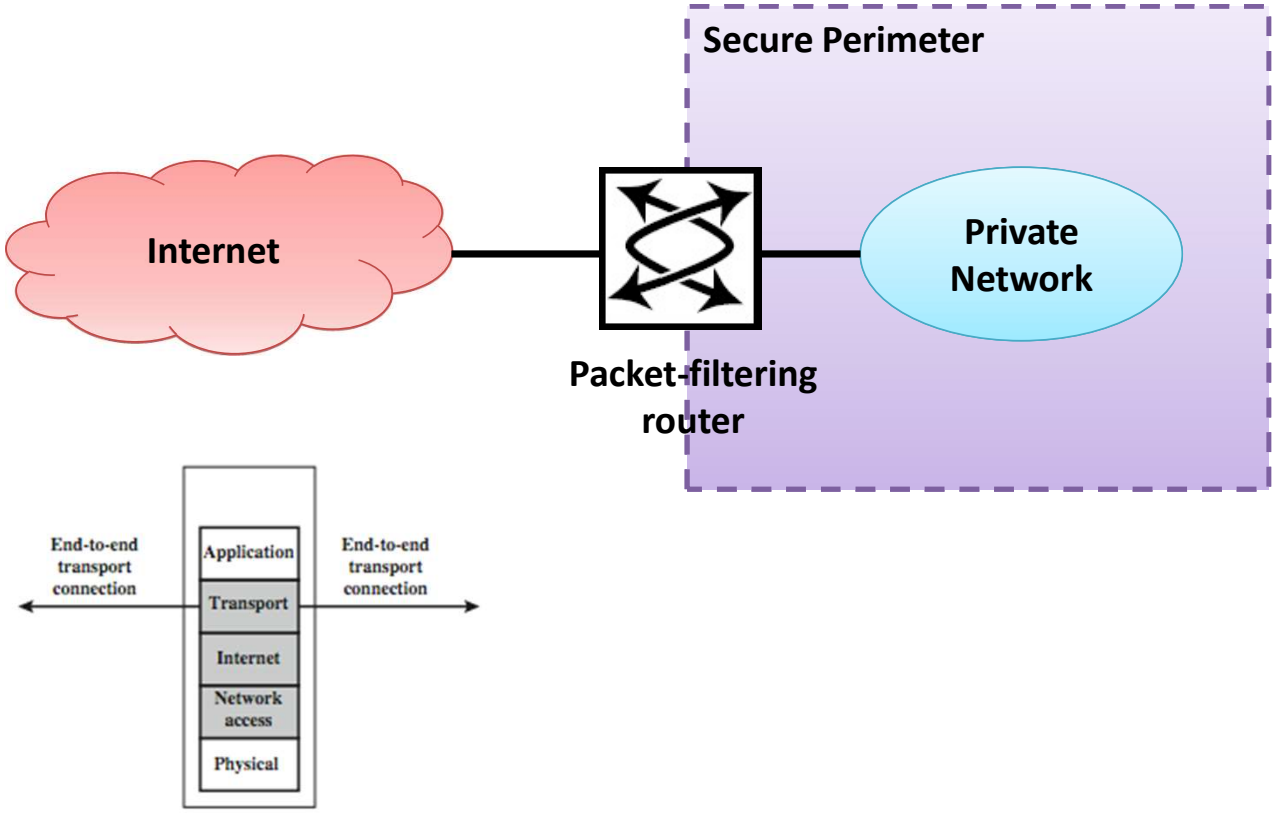
Protection Methods

- **Packet Filtering**
  - Rejects unauthorised TCP/IP packets or connection attempts.
- **Application-Level Gateway**
  - Acts as a relay of application-level traffic.
- **Circuit-Level Gateway**
  - AKA Network Address Translation (NAT)
  - Translates the addresses of internal hosts in order to hide them from the outside world.
- **Proxy Services**
  - Makes high level application level connections to external hosts on behalf of the internal hosts to completely isolate the internal form external hosts.

Other common Firewall Services

- Encrypted Authentication
- VPN to avoid expensive leased lines
- Virus Scanning
- Content Filtering (added to the proxy server)
- Other ..

Packet Filtering Router



Packet-filtering router

- Filters the IP packets, forwarding or discarding them depending on a list of rules.
- The rules are based on IP fields and transport header (TCP, UDP).
- Filters packets in both directions.

Packet-filtering router

Action	Outhost	Port	Theirhost	Port	comment
Block	*	*	Dave	*	We don’t trust him
Allow	OUR-GW	25	*	*	Connection to our mailer port
:	:	:	:	:	:

Packet-filtering router

- **Disadvantages**:
  - Difficulty setting up rules and no authentication.
  - IP addresses of hosts on the protected side of the filter can be readily determined by observing the packet traffic on the unprotected side of the filter
  - Filters cannot check all of the fragments of higher level protocols (like TCP) as the TCP header information is only available in the first fragment.
  - Filters are not sophisticated enough to check the validity of the application level protocols imbedded in the TCP packets

Packet-filtering router

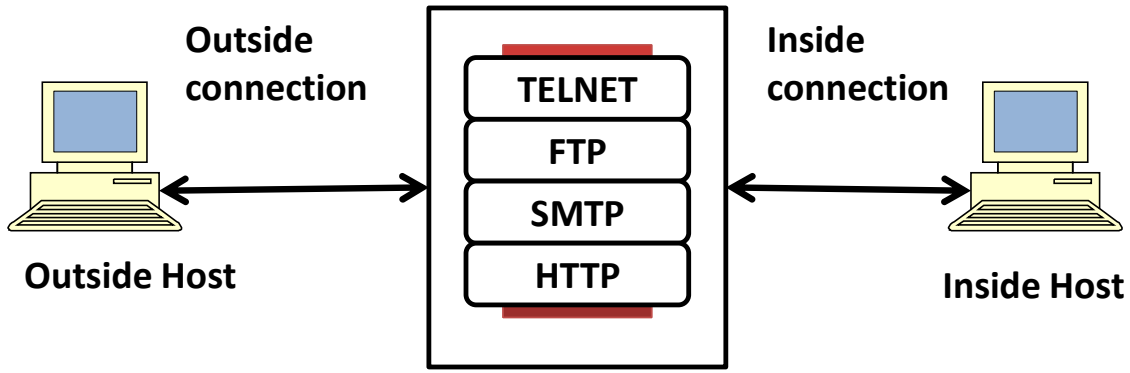
- **Some attacks:**
  - **IP address spoofing**
    - Fake resource address to be trusted

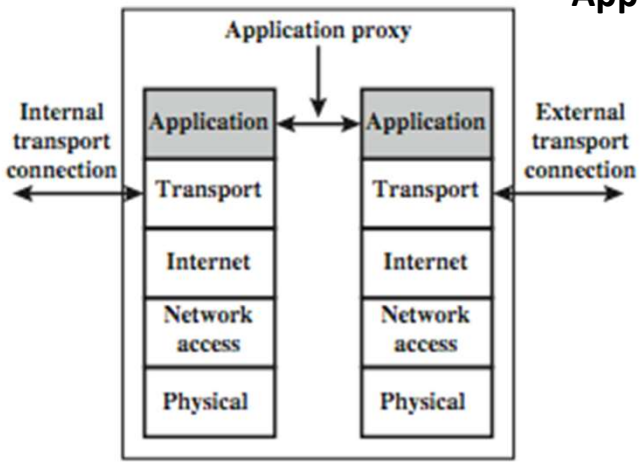
    →Add filters on router to block
  - **Source routing attacks**
    - Attacker sets a route other than default

    → Block source routed packets
  - **Tiny fragments attacks**
    - Split header info over several tiny packets

    → Either discard or reassemble before check

Application-level Gateway





Application-level Gateway

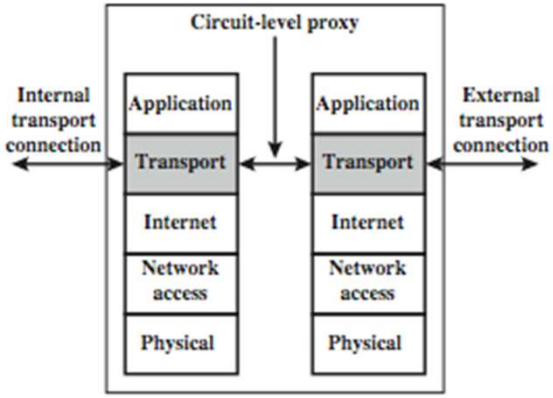
- An application-level gateway (or proxy server), acts as a relay of application-level traffic.
- A user contacts the gateway to access some service, provides details of the service, remote host & authentication details, contacts the application on the remote host and relays all data between the two endpoints.
- If the gateway does not implement the proxy code for a specific application, then it is not supported and cannot be used.
- Some services naturally support proxying, whilst others are more problematic.
- Application-level gateways tend to be more secure than packet filters, &can log and audit traffic at application level.

Application-level Gateway

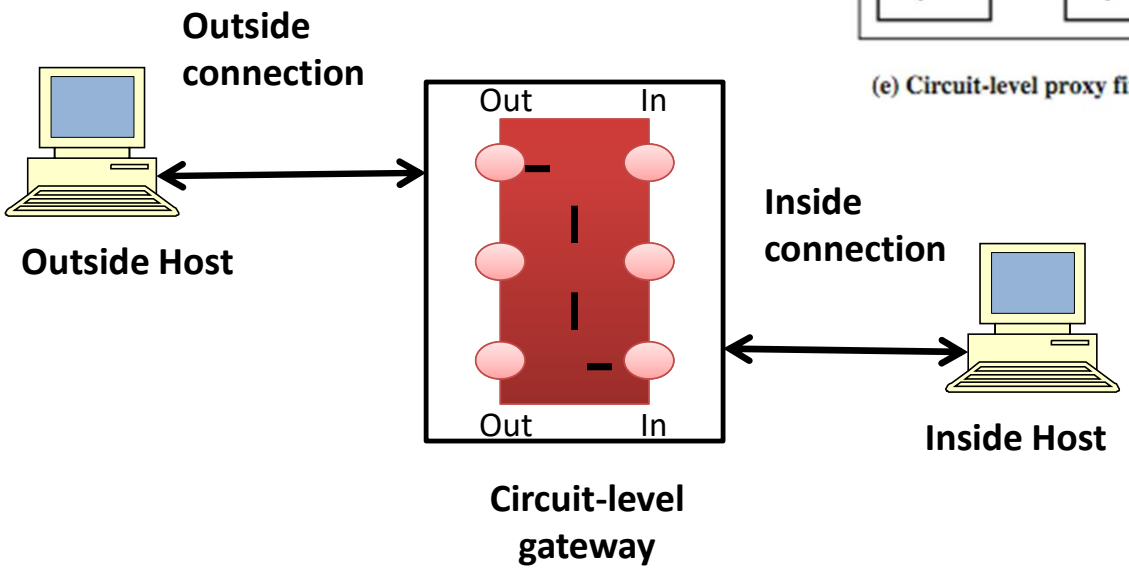
- They are more secure than packets filters as only scrutinise a few allowable applications.

- Disadvantage:
  - Additional processing overhead for each connection.

Circuit-level Gateway



(e) Circuit-level proxy firewall



Circuit-level Gateway

- It does not permit an end-to-end TCP connection, but rather relays them.
- It opens two connections between
  - Itself and inner host
  - Itself and outside host
- Security function consist of determining which connections are allowed.

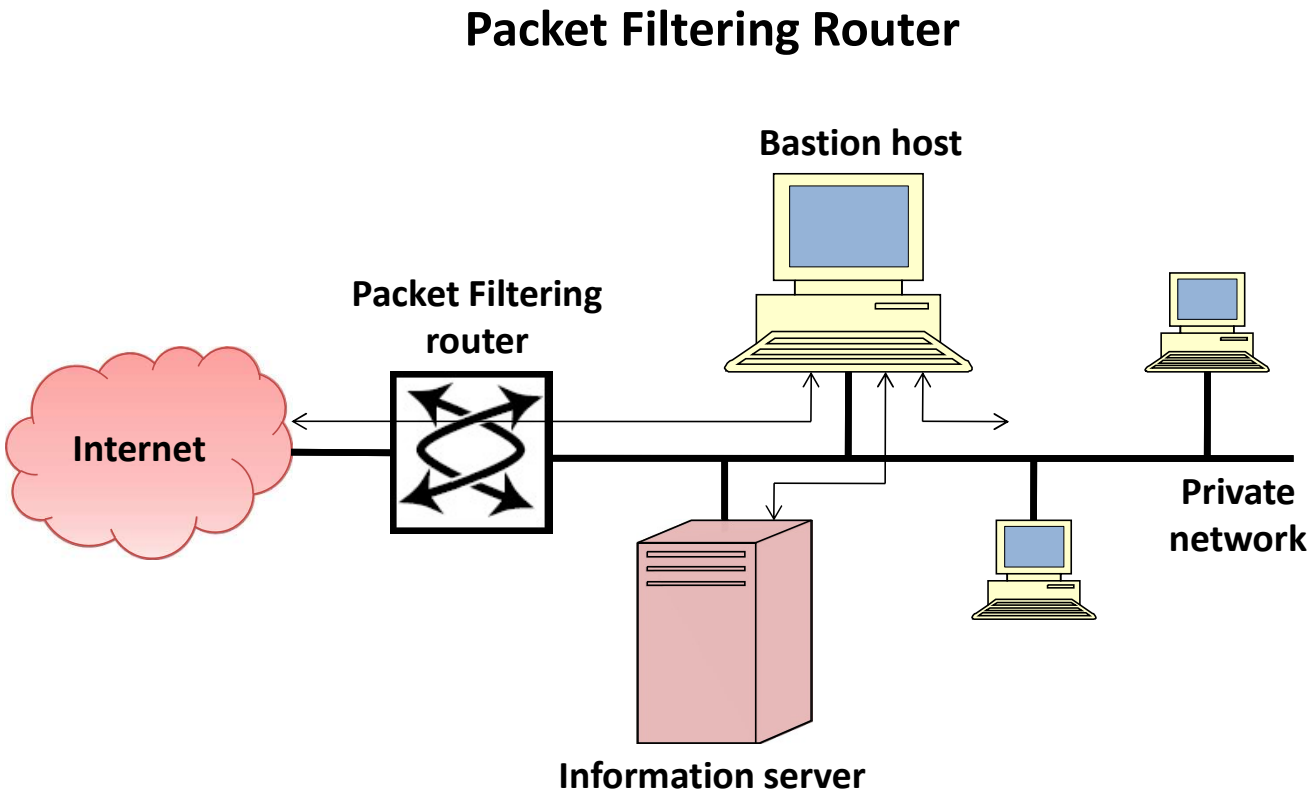
Bastion Host

- Is a critical strong point in the network's security.
- It serves as a platform for application–level or circuit–level gateway.
- Its hardware executes a secure version of its operating system (trusted system)
- Before the user is allowed access the bastion host can require authentication of the user
- Only "essential" services are installed, like proxy Telnet, DNS, FTP, . . .

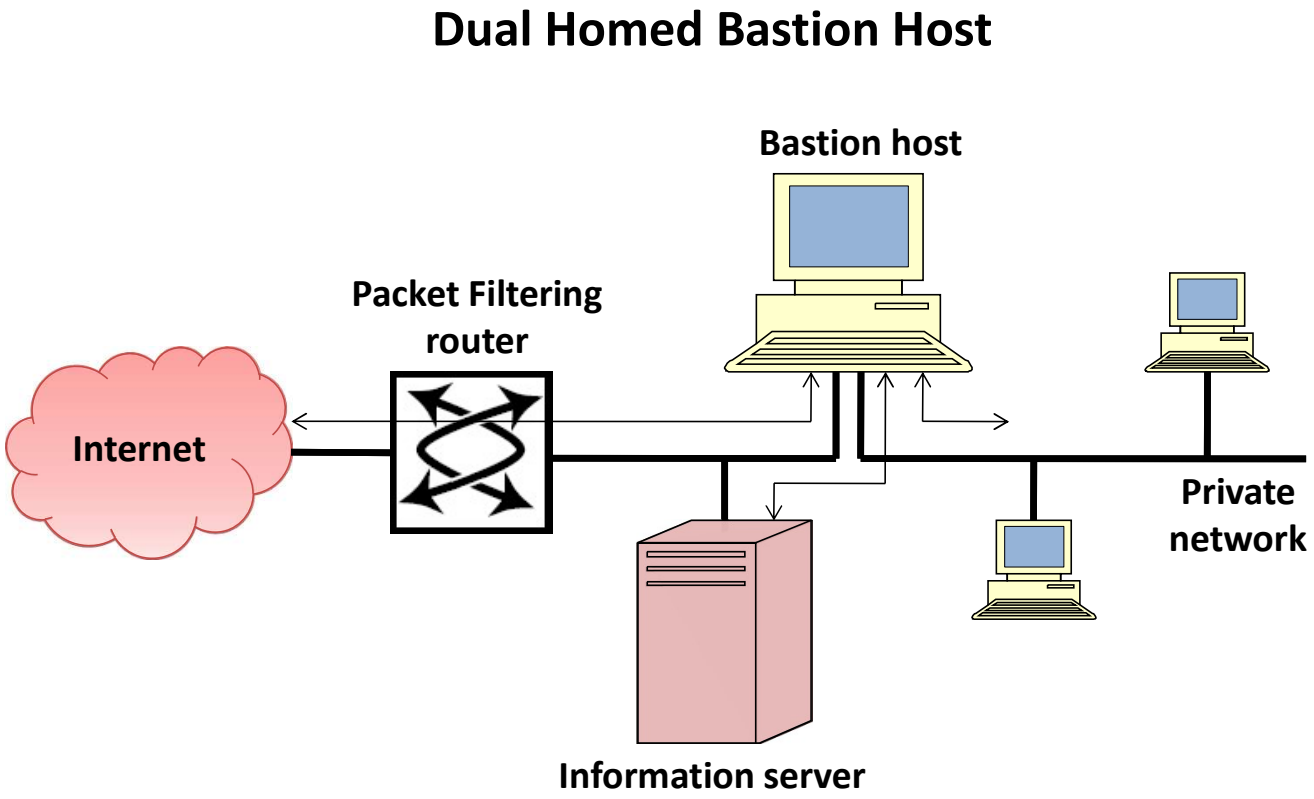
Bastion Host

- **The proxy**
  - Supports only a subset of the application's command set
  - Access to only some of the host systems
  - Keeps audits
  - Small software package
  - Independent of other proxies
  - No disk access
  - Run as non-privileged user

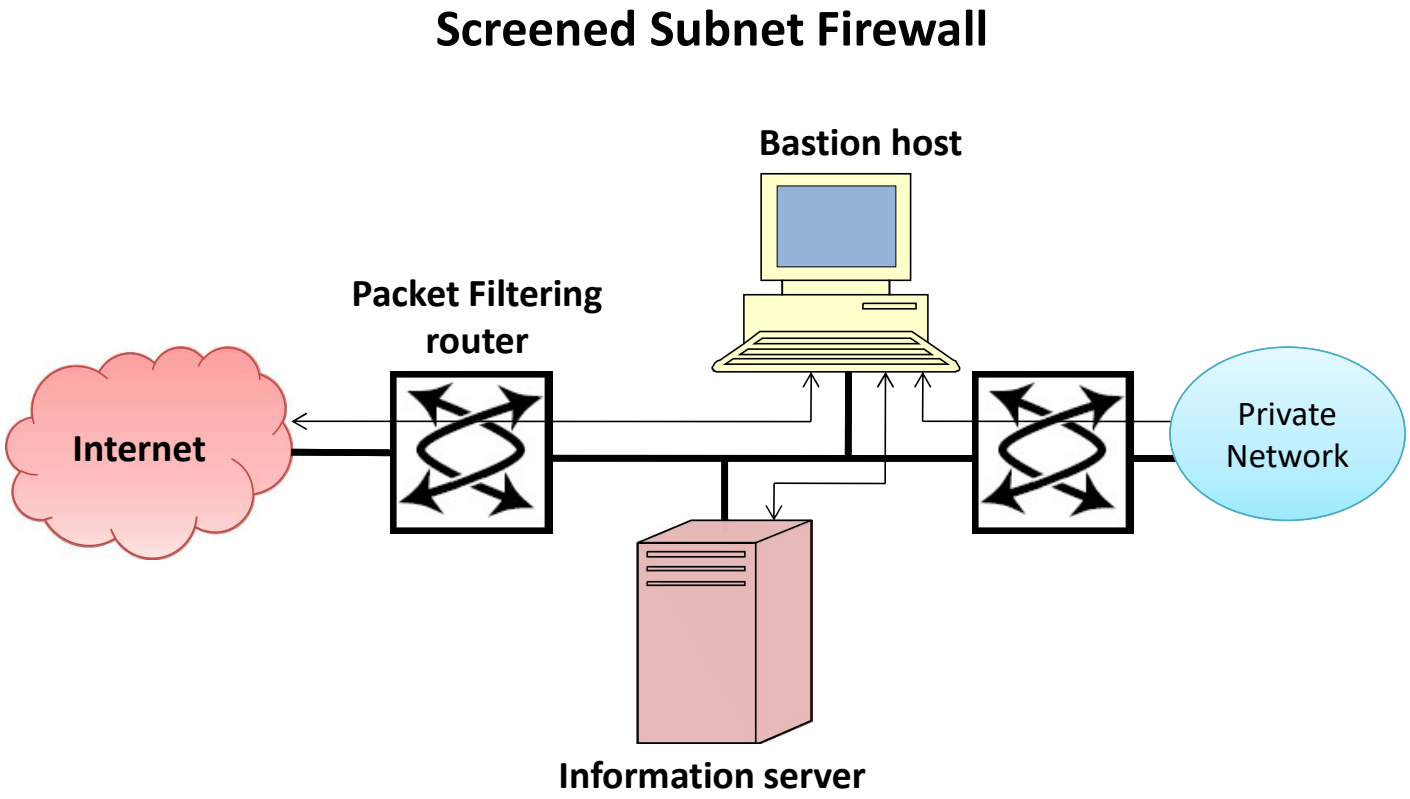
Firewall configuration



Firewall configuration

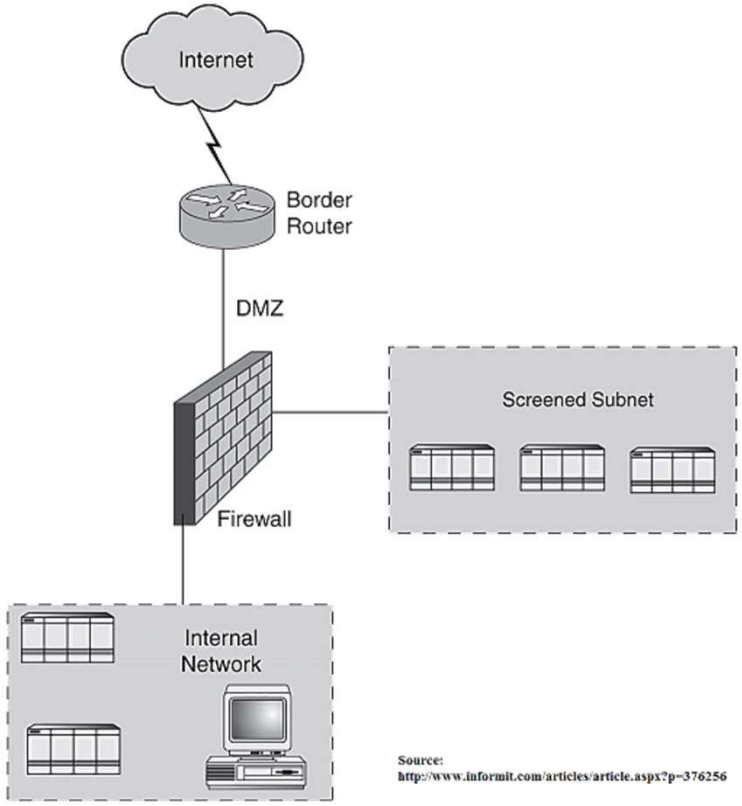


Firewall configuration

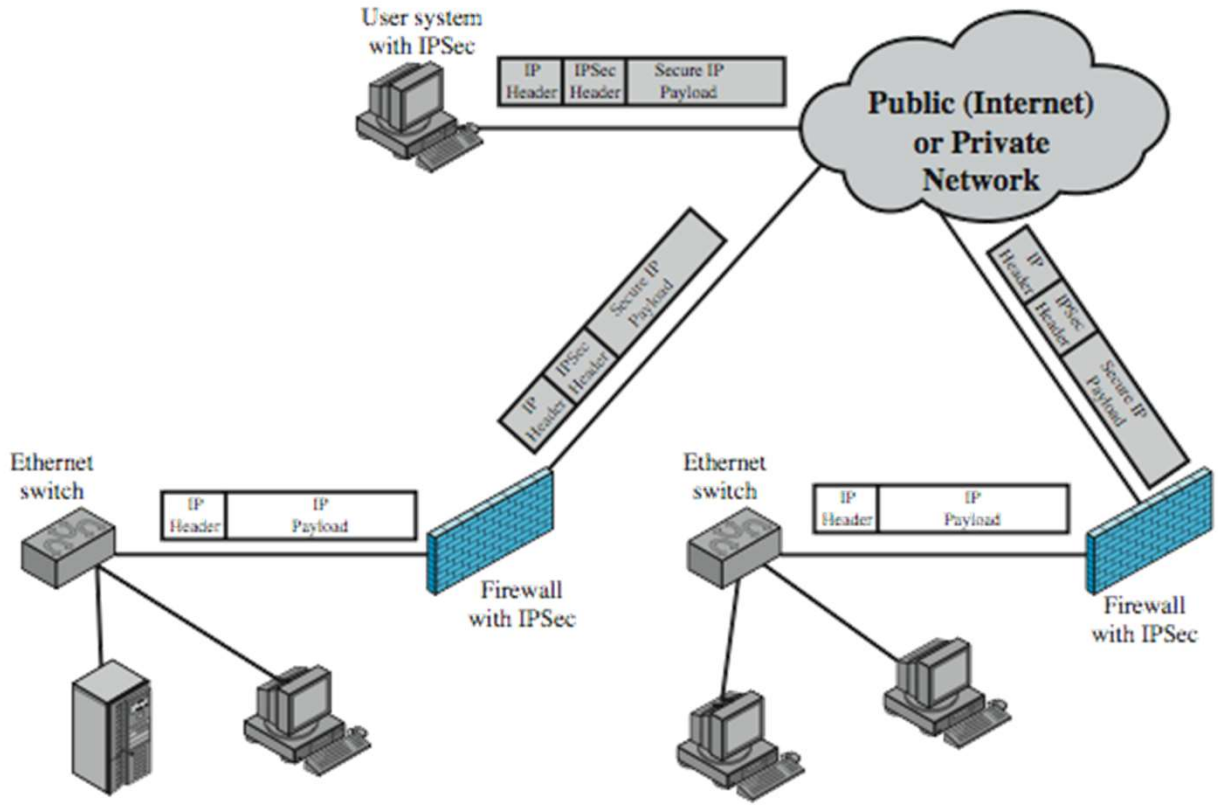


Firewall Configurations

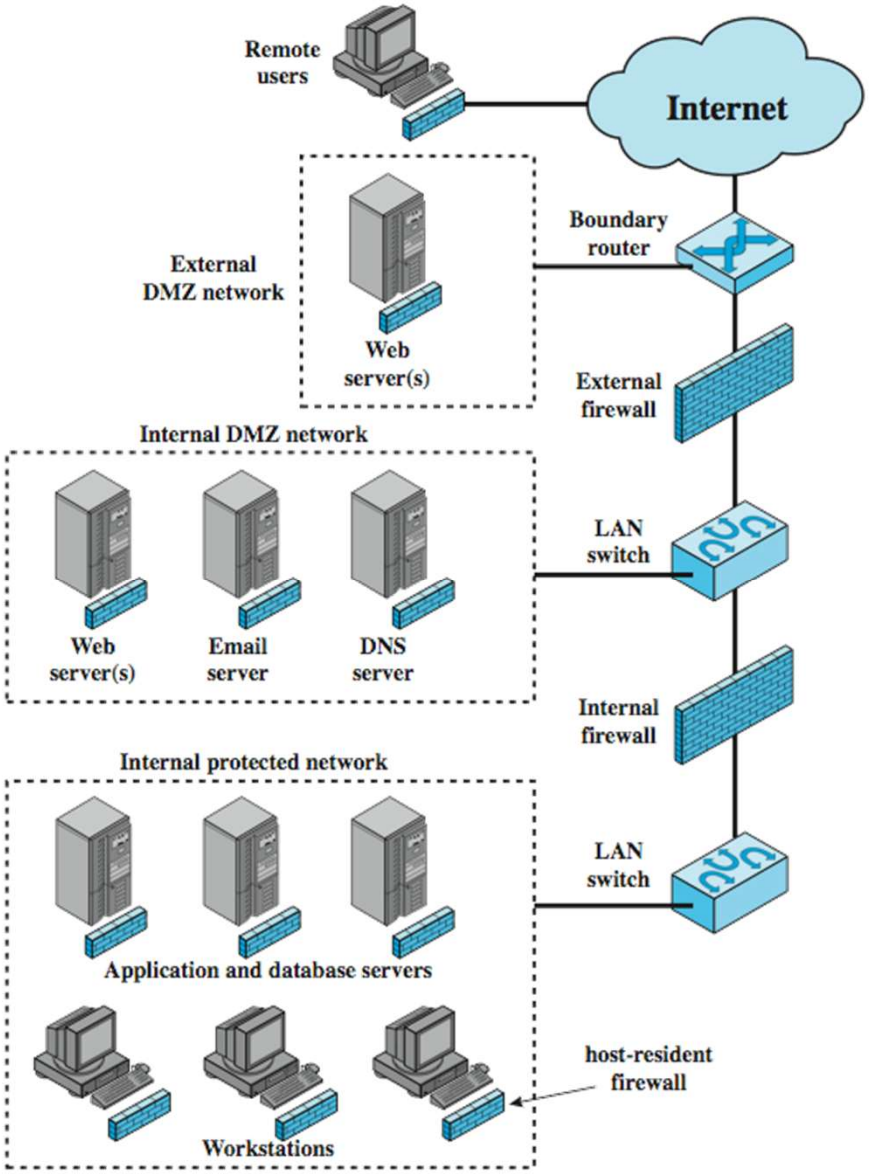
- De-Militarised Zones and Screened Subnets
  - DMZ and screened subnet refer to a small network containing public services connected directly to and offered protection by the firewall or other filtering device.
  - A firewall or a comparable traffic-screening device protects a screened subnet that is directly connected to it.
  - A DMZ is in front of a firewall, whereas a screened subnet is behind a firewall.



Virtual Private Networks



Distributed Firewalls



Bastion Host and Trusted Systems

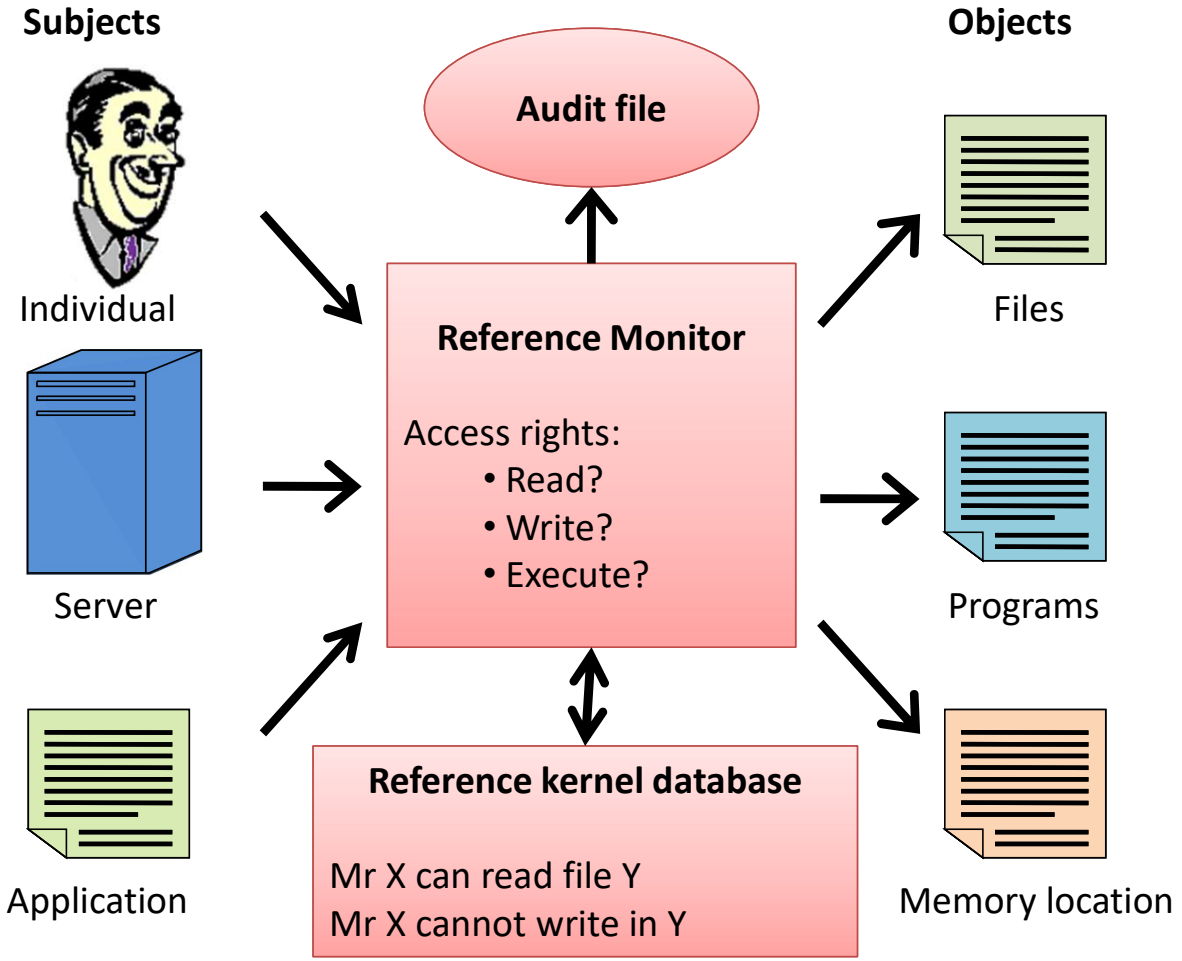
- To protect data or resources on the basis of level security (confidential, secret, top secret, ultra . .)

- Example
  - In a company, the strategic planning is only available to corporate officers and the financial data only accessible to administration officers.

Trusted Systems: Multilevel Security

- The system enforces the following rules
  - No read up (simple security property): A subject can only read an object of less or equal security level.
  - No write down (*–property): A subject can only write into an object of greater or equal security level.

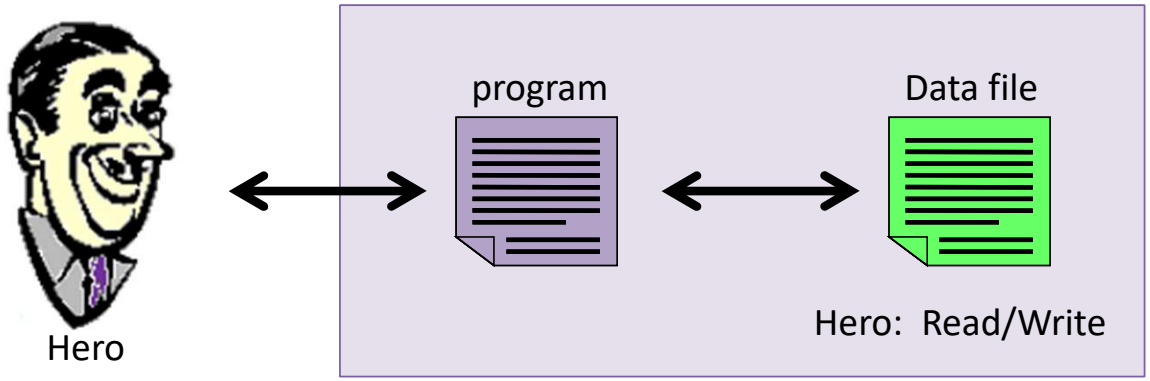
Reference Monitor Concept



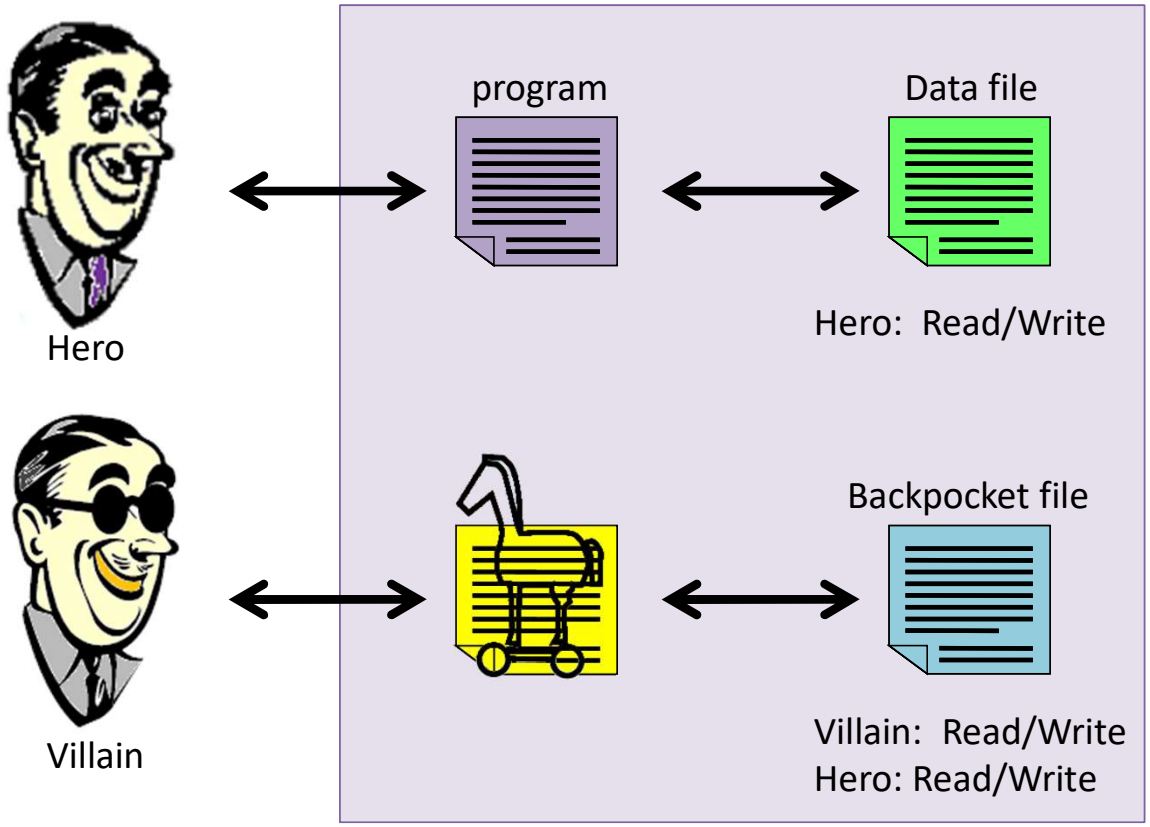
Trusted Systems

- Properties of the Reference Monitor
  - Complete mediation.
  - Isolation.
  - Verifiability.

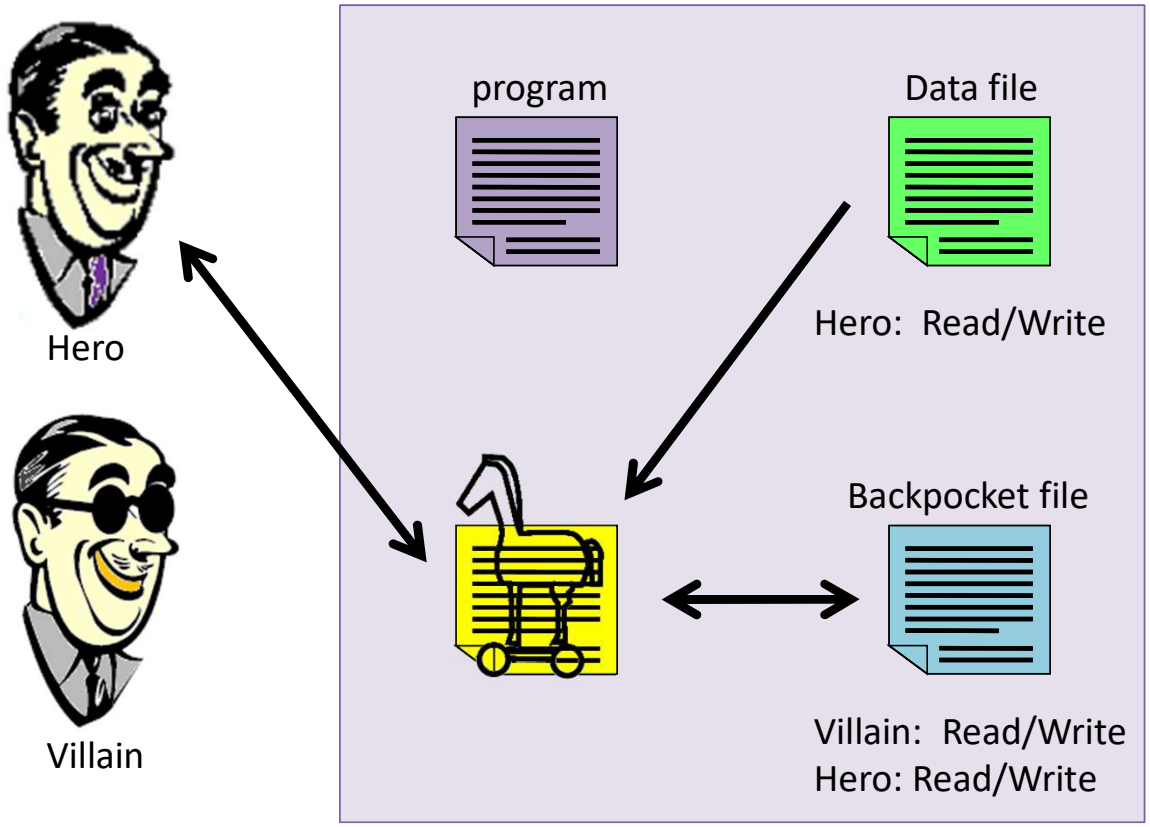
Example: Trojan Horse



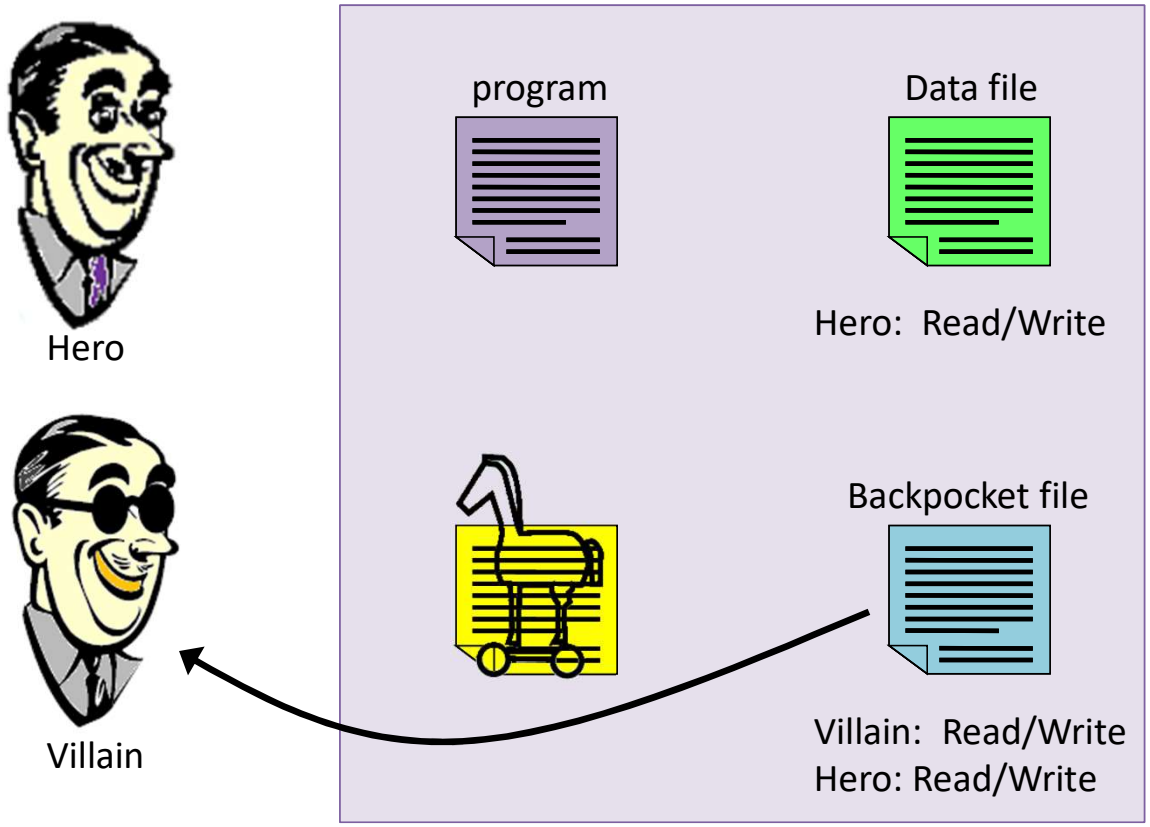
Example: Trojan Horse



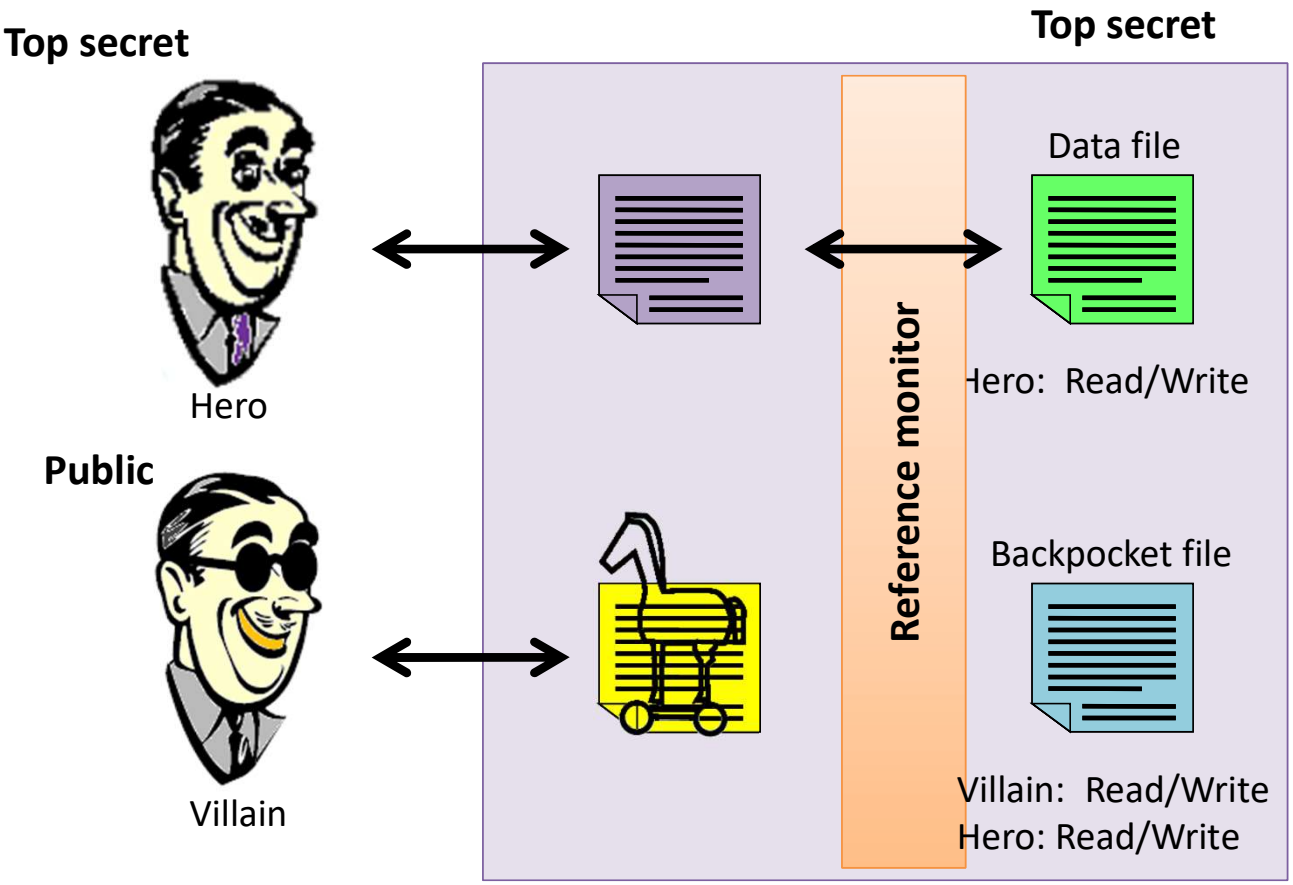
Example: Trojan Horse



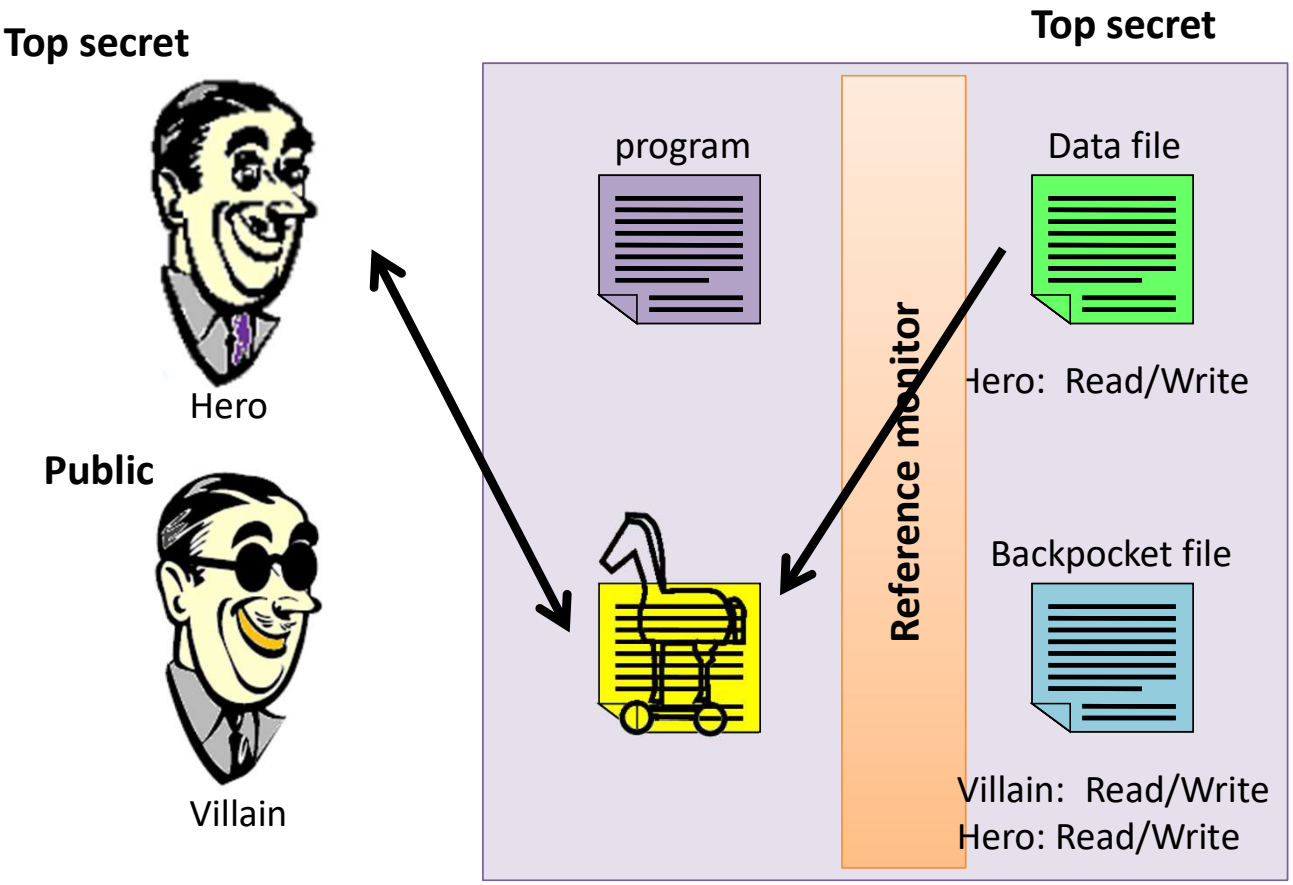
Example: Trojan Horse



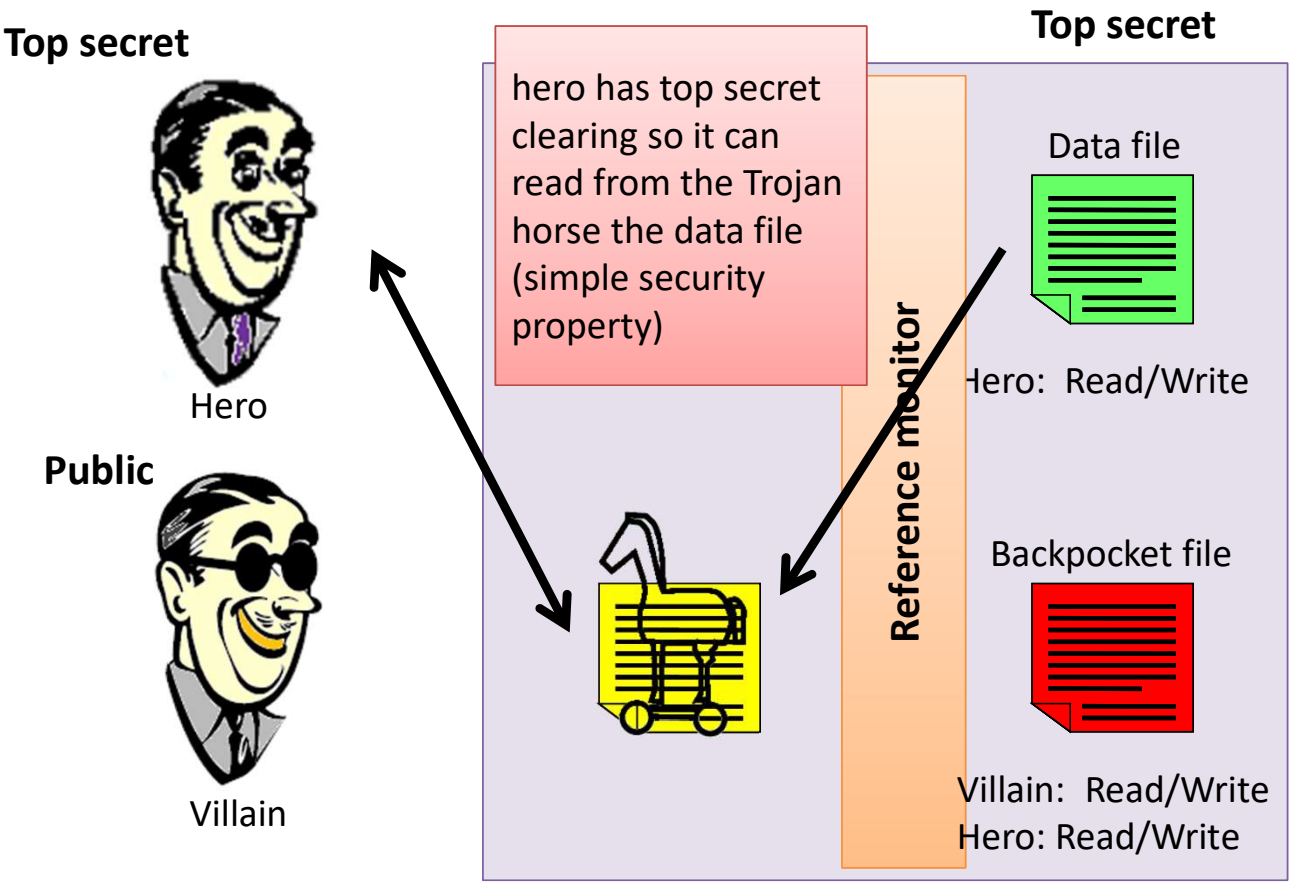
Example: Trojan Horse



Example: Trojan Horse



Example: Trojan Horse



Summary

- The Firewall is inserted between the premises network and the Internet.
- Firewall also provides
  - Service Control
  - Direction Control
  - Use Control
  - Behaviour Control
- The Firewall cannot protect against attacks that bypass the Firewall.
- Some attacks:
  - IP address spoofing
  - Source routing attacks
  - Tiny fragments attacks
- Trusted Systems
- Trojan Horse attacks