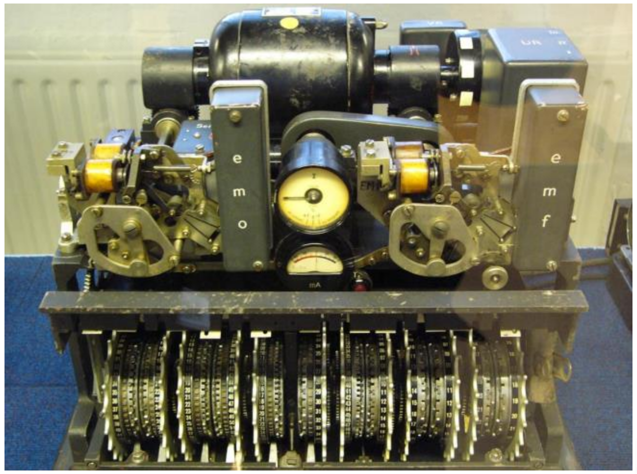
CONNECTION SECURITY



The German Lorenz SZ42 Cipher Machine at the National Museum of Computing (TNMOC), Bletchley Park, UK. The SZ42 was used by the German High Command in the early 1940s to encrypt telecommunication messages during World War II.

Contents

Part I

- Crypto applied
  - Choice of algos
  - Object security
  - Perfect Forward Secrecy
  - Opportunistic encryption
  - Forward/Backward security

Part II

- Access
  - Triple A (AAA)
    - CHAP, EAP
    - Radius, Diameter
  - SSO, FID

Part III

- Secure connections
  - Internet encryption
  - TLS, DTLS
  - IPSEC
  - Internet of Things (IoT)
  - GBA
  - GSM, LTE

Part IV

- Threats
  - BOTNETS
  - (D)DOS

This lecture is about

- Security in some major data transport protocols
- Understanding crypto usage in network protocols
- Access Control to network services and
- How we can protect data that is sent through a network against
  - passive eavesdropping
  - active eavesdropping
    - insert messages
    - modify(substitute) messages
    - replay messages
    - change/spoof origin of data

Mandatory reading

- Comparison between RADIUS and Diameter, A Hosi, HUT, Finland, 2003.
- Cryptography in an all encrypted world , C. Jost, et al, Ericsson Review, Dec 2015.
- Application Layer Security for the Internet of Things, G Selander, F Palombini, J Mattsson, L Seitz, unpublished (read project B description how to access this file)
- Understanding the Mirai Botnet, M Antonakakis, et all, Usenix, Aug. 2017.
- Security in the Evolved Packet System, R. Blom, et al. Ericsson Review, Oct 2010

PART I

CRYPTO APPLIED

Mandatory reading:

Cryptography in an all encrypted world , Ericsson Review, Dec 2015

Refreshing of some facts

- What kind of security functions are needed in secure network protocols?
- Where to place security functions ?
  - High or low in the stack, hardware or software?
- What are important criteria for choosing crypto algorithms for certain tasks.
- Where do the keys come from and how to handle them?
- What happens if we can do quantum computing?

Location of crypto

High in the stack

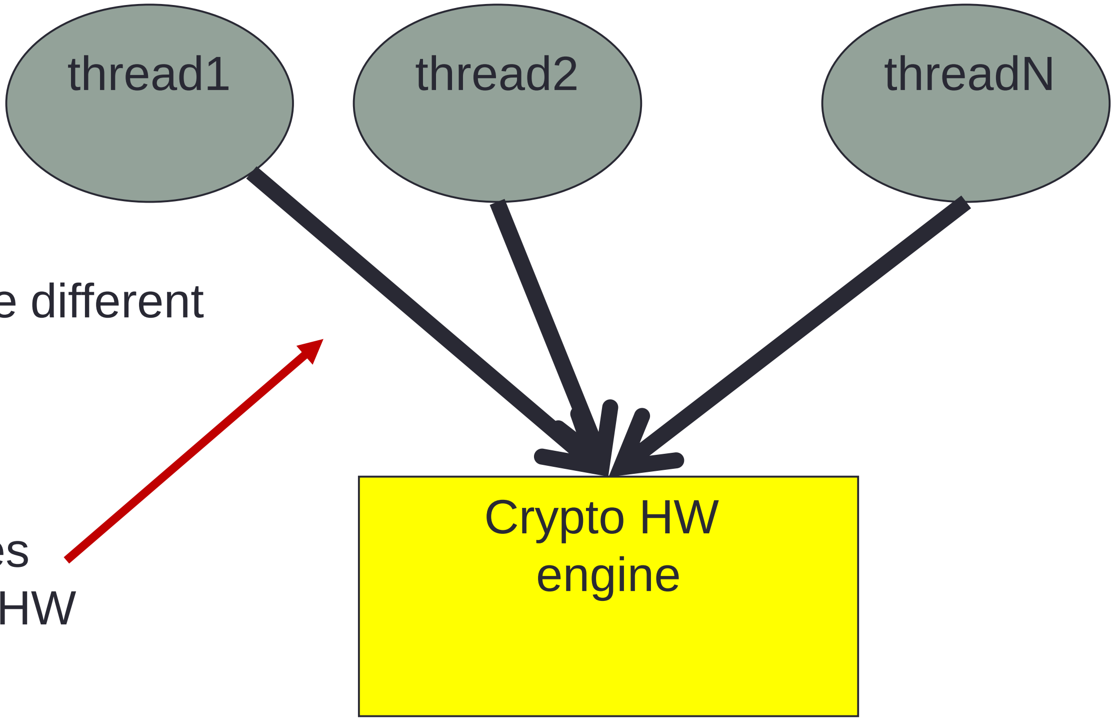
Not complex
Adapted to application

often complex
Tradeoffs to handle different use cases

Not complex
can be fast, requires efficient sharing of HW

Low in the stack

HW crypto or SW



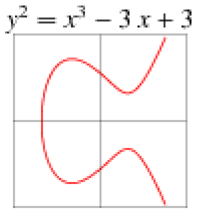
Example: Choice of signature algo

- Security: how strong protection needed?
- Do we really need non-repudiation in my case?
- Message size restrictions?
- Key size (storage) restrinctions?
- Compliance to standards

- Example: RSA vs ECC

- Remark: signing of documents and certificates has higher security demands because the life-time of the signature is long. Compare this to use cases were the signature only needs to be secure a limited time-span.

Performance: RSA vs ECC (1/2)



http://nicj.net/files/performance_comparison_of_elliptic_curve_and_rsa_digital_signatures.pdf

- Key size

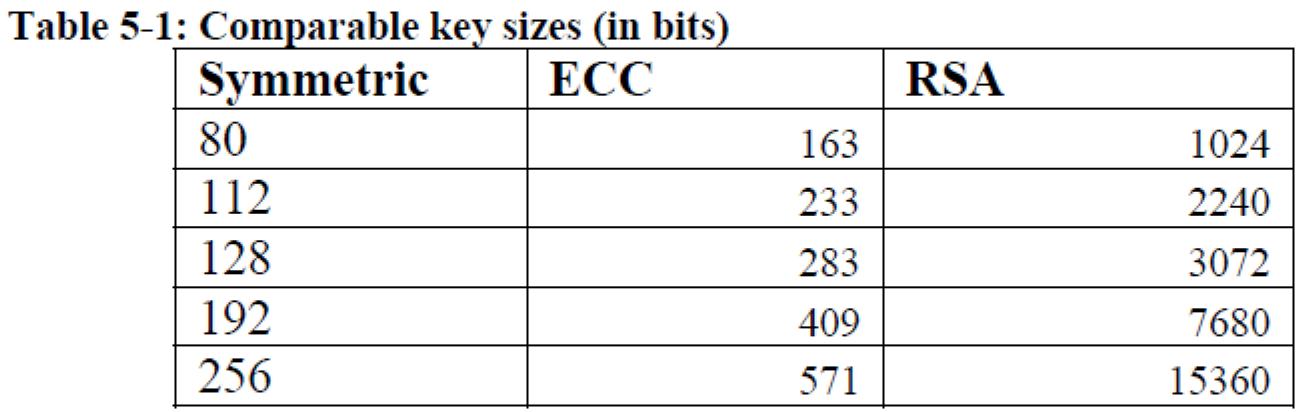


Table 5-1: Comparable key sizes (in bits)

Symmetric	ECC	RSA
80	163	1024
112	233	2240
128	283	3072
192	409	7680
256	571	15360

- Generation

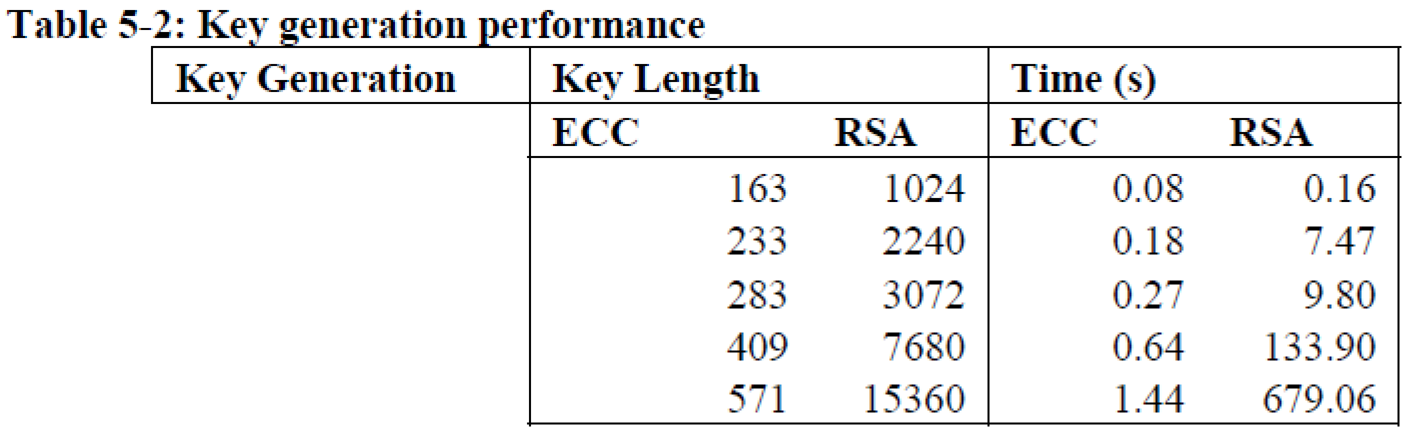


Table 5-2: Key generation performance

Key Generation	Key Length		Time (s)	
	ECC	RSA	ECC	RSA
	163	1024	0.08	0.16
	233	2240	0.18	7.47
	283	3072	0.27	9.80
	409	7680	0.64	133.90
	571	15360	1.44	679.06

Performance: RSA vs ECC (2/2)

http://nicj.net/files/performance_comparison_of_elliptic_curve_and_rsa_digital_signatures.pdf

- Signature generation

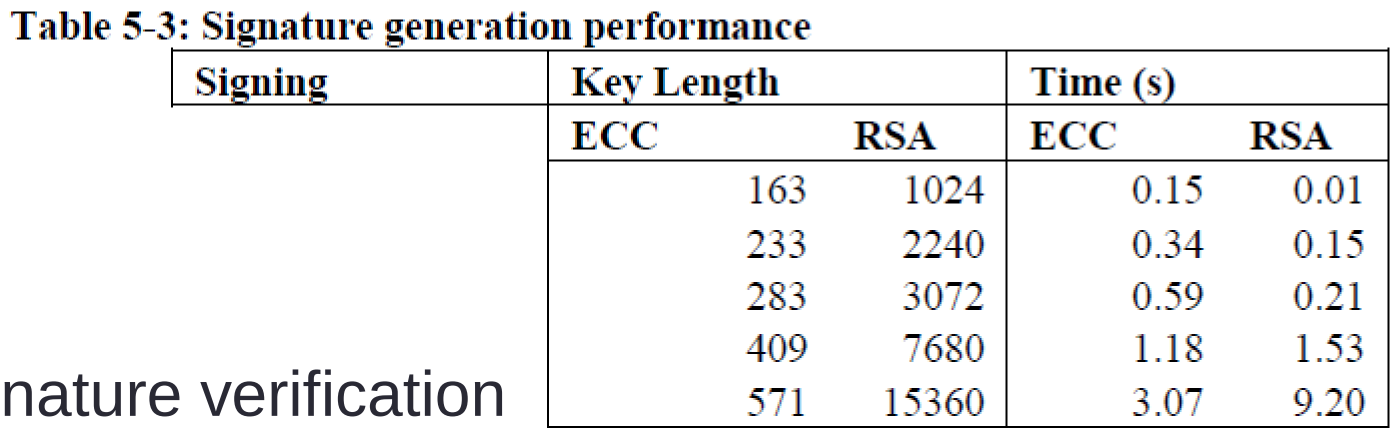


Table 5-3: Signature generation performance

Signing	Key Length		Time (s)	
	ECC	RSA	ECC	RSA
	163	1024	0.15	0.01
	233	2240	0.34	0.15
	283	3072	0.59	0.21
	409	7680	1.18	1.53
	571	15360	3.07	9.20

- Signature verification

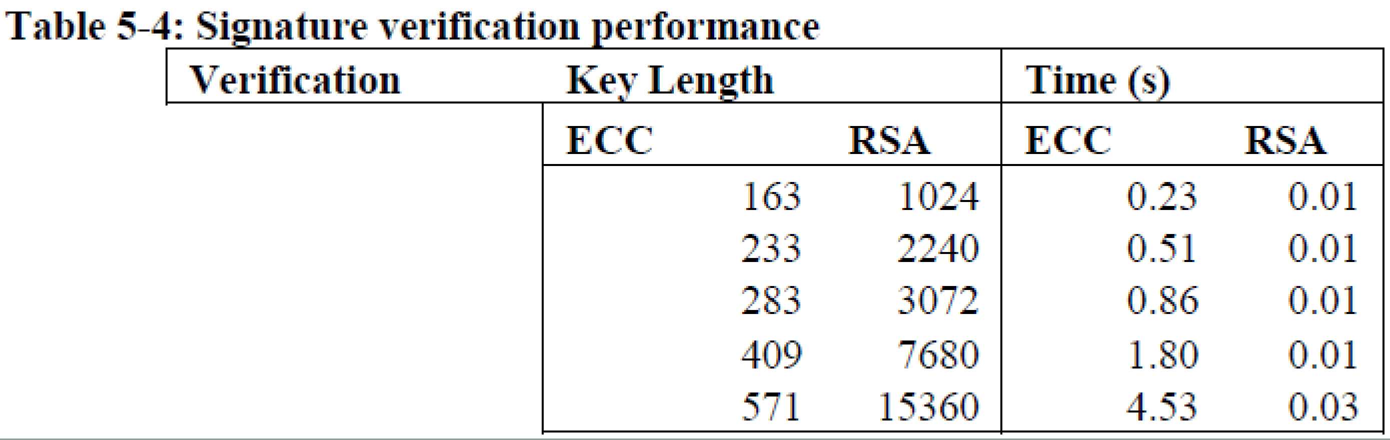


Table 5-4: Signature verification performance

Verification	Key Length		Time (s)	
	ECC	RSA	ECC	RSA
	163	1024	0.23	0.01
	233	2240	0.51	0.01
	283	3072	0.86	0.01
	409	7680	1.80	0.01
	571	15360	4.53	0.03

ECC: what to pick

Obtaining the RSA parameters is very simple but how to choose an ECC is more problematic.

- Good choice for signatures: The Elliptic Curve Digital Signature Algorithm (ECDSA) is defined in FIPS 186-2 as a standard for US government digital signatures, and described in ANSI X9.62. Or the Brainpool curves.
- Stay away from curves over finite fields with characteristic 2 unless you have your IPR situation covered. (still valid)

Key handling/management

- **Where do the crypto keys come from ?**
  - OoB (Out of Band)
  - In band:
    - a) by subprotocol or
    - b) key management protocol

- **Where are the keys stored ?**
  - During sessions
  - Between sessions
  - Life-time:
    - **ephemeral** keys (short-lived) vs master keys

Example: key entropy erosion

256 bit key MAC scheme with tag of size 128 bit

Suppose our MAC is well-behaved, then because tag size is 128 bit : $P_I = 2^{-128}$

Hence I(K; M,T) = 128 bit

which implies that that the observation of (m,t) will reduce the initial key entropy of 256 bit by 128 bit to 128 bit.

SO: why does MAC protection work in practice even after many messages being sent???

Important ‘new’ things in crypto

- (Perfect) Forward Secrecy
- Forward and Backward Security
- Opportunistic encryption
- Object security
- Post-quantum cryptography

(Perfect) Forward Secrecy (PFS)

Perfect Forward Secrecy is a feature of a key agreement protocols that gives assurances your session keys, even those in the past, will not be compromised even if the private master key is compromised.

Note: But when the crypto scheme is broken they might be lost

Weak Perfect Forward Secrecy is like PFS but secrecy of session keys is only guaranteed for sessions in which the ***attacker did not participated***.

Most modern protocols like TLS 1.2/1.3 have support for PFS

Forward & Backward Security

- **Forward Security:**
  - Secrecy of new session keys is regained even if a previous session key(s) are compromised

- **Backward Security:**
  - Secrecy of old session keys is maintained when a new session key is lost.

Session keys

Opportunistic Encryption

- A system does Opportunistic Encryption (OE) if it always tries to encrypt the traffic after it connects **even if the other party cannot be authenticated**. If no encryption is possible one either falls back to a non-encrypted communications or the connection is dropped (depends on a policy/design).

- This protects against passive eavesdropping.

- Snowden affair increased popularity of OE

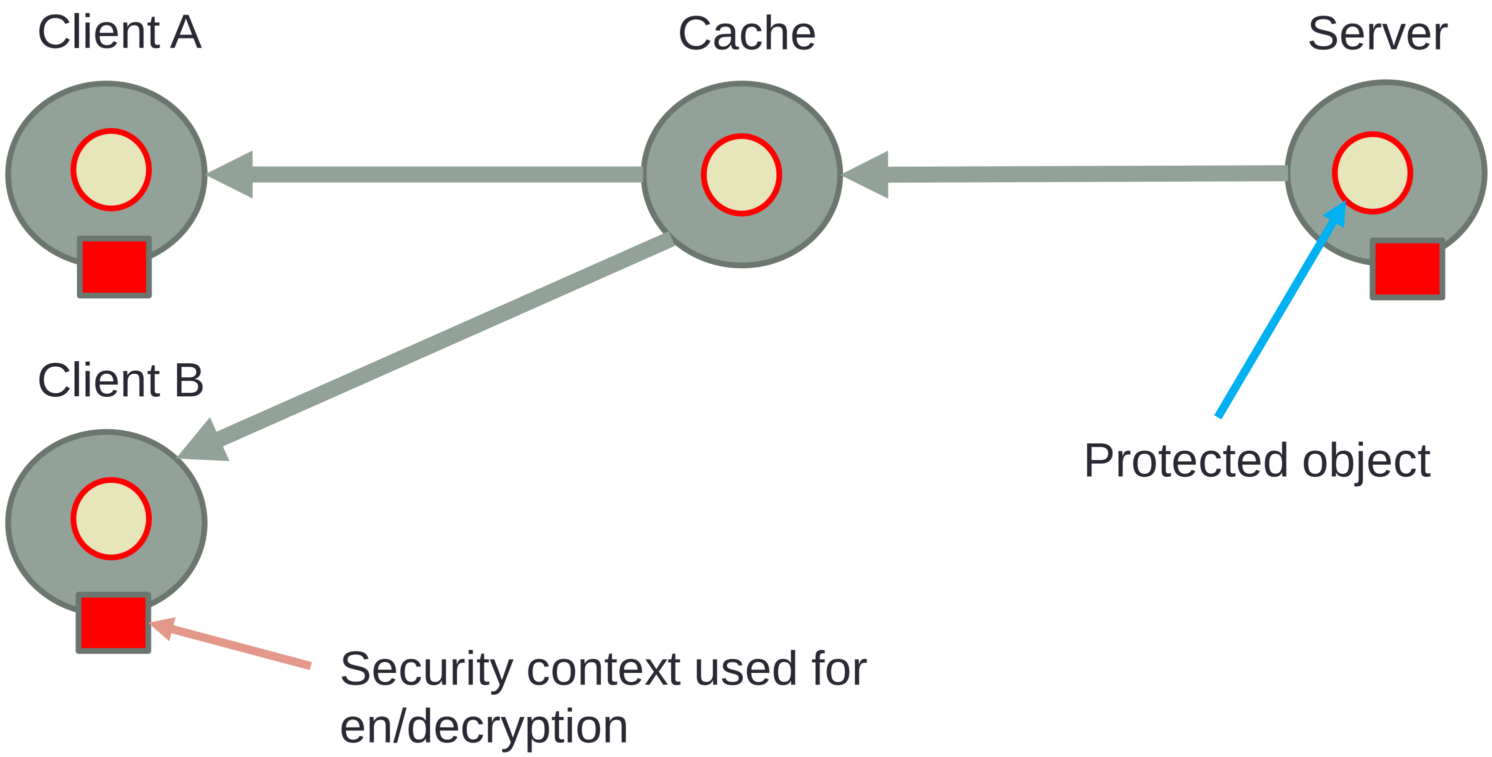
Object security

- Instead of encrypting the data channel one can encrypt the data before it is sent. Then the data is still considered to be an object and one uses the term object security
  - Advantage: encrypted objects can be cached in the network to increase network performance. No or very loose session maintenance. Can be one-directional or broadcasted
  - Disadvantage(s): increase of overhead (more bits to be sent). Less easy to combine data from different sources

See: Object Security in Web of Things, J Mattsson, G Selander, 2014, W3 Org.

Note: (D)TLS and IPsec do not result in object security

Object security – we can cache



Post Quantum Cryptography (PQC)

Will quantum computing break all crypto?

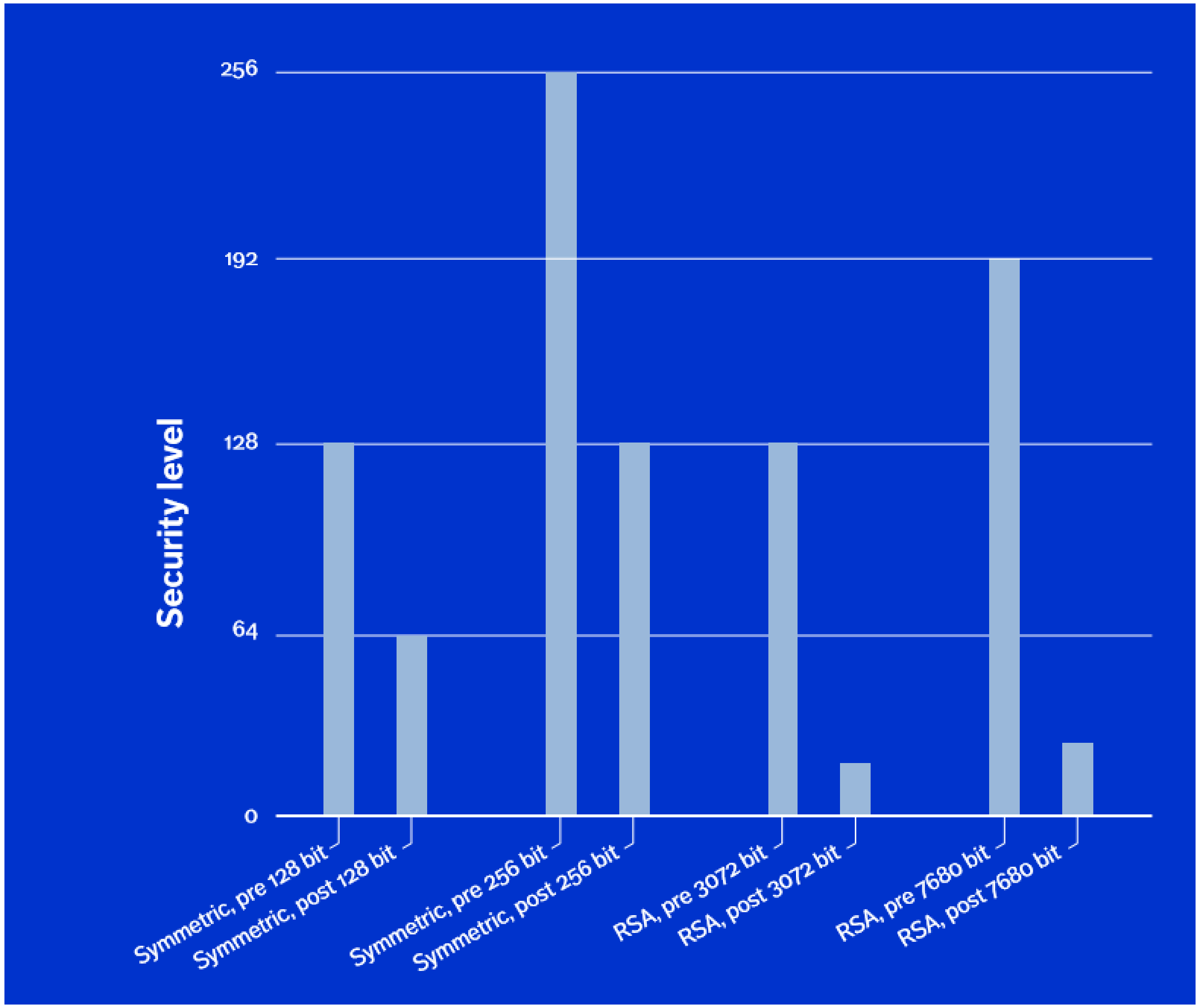
Answer: no.

- The basic message is that symmetric encryption schemes have to double their key size (due to Grover's algorithm)
- Hashes are ok or maybe also need to double size
- Public-key crypto: many schemes break due to Shor's algorithm, especially the ones that are today used in TLS and Ipsec.
- Shor's algorithm is considered to be very realistic to implement.

Background reading from ETSI workshop

- http://www.etsi.org/images/files/Events/2014/201410_Crypto/e-proceedings-QSC-14.pdf

Some figures



From Cryptography in an all encrypted world , Ericsson Review, Dec 2015

PQC Crypto

- There is a lot of activity to develop crypto algorithms that can even be used when quantum computing is possible.

- Hash functions
  - Not much of problem: increase the block size
- Symmetric key encryption:
  - Not so much of a problem. Double the key size if needed.

- Asymmetric key encryption (digital signatures)
  - Essentially all existing schemes need to be replaced

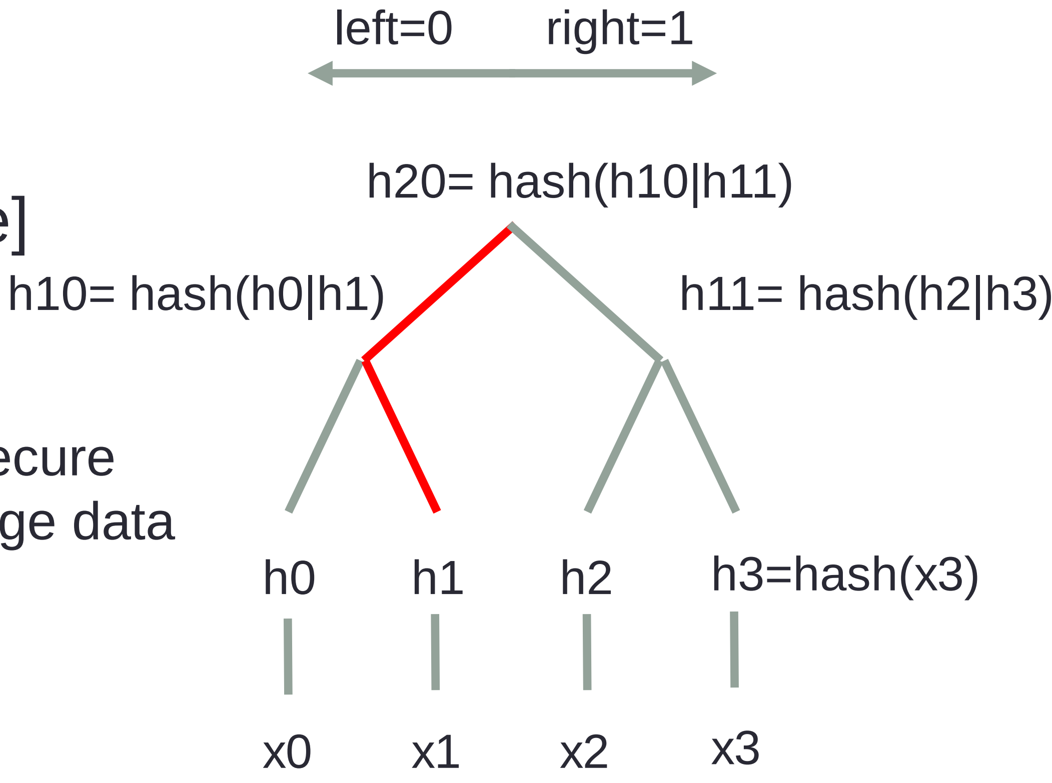
Merkle Tree based scheme

- One approach to devise PQC signatures follows an idea of Merkle and uses a hash tree.

- Fingerprint of x1:

  [h20, left-right path in tree]

Hash trees allow efficient and secure verification of the contents of large data structures.



Used in product called KSI of Guardtime

PART II

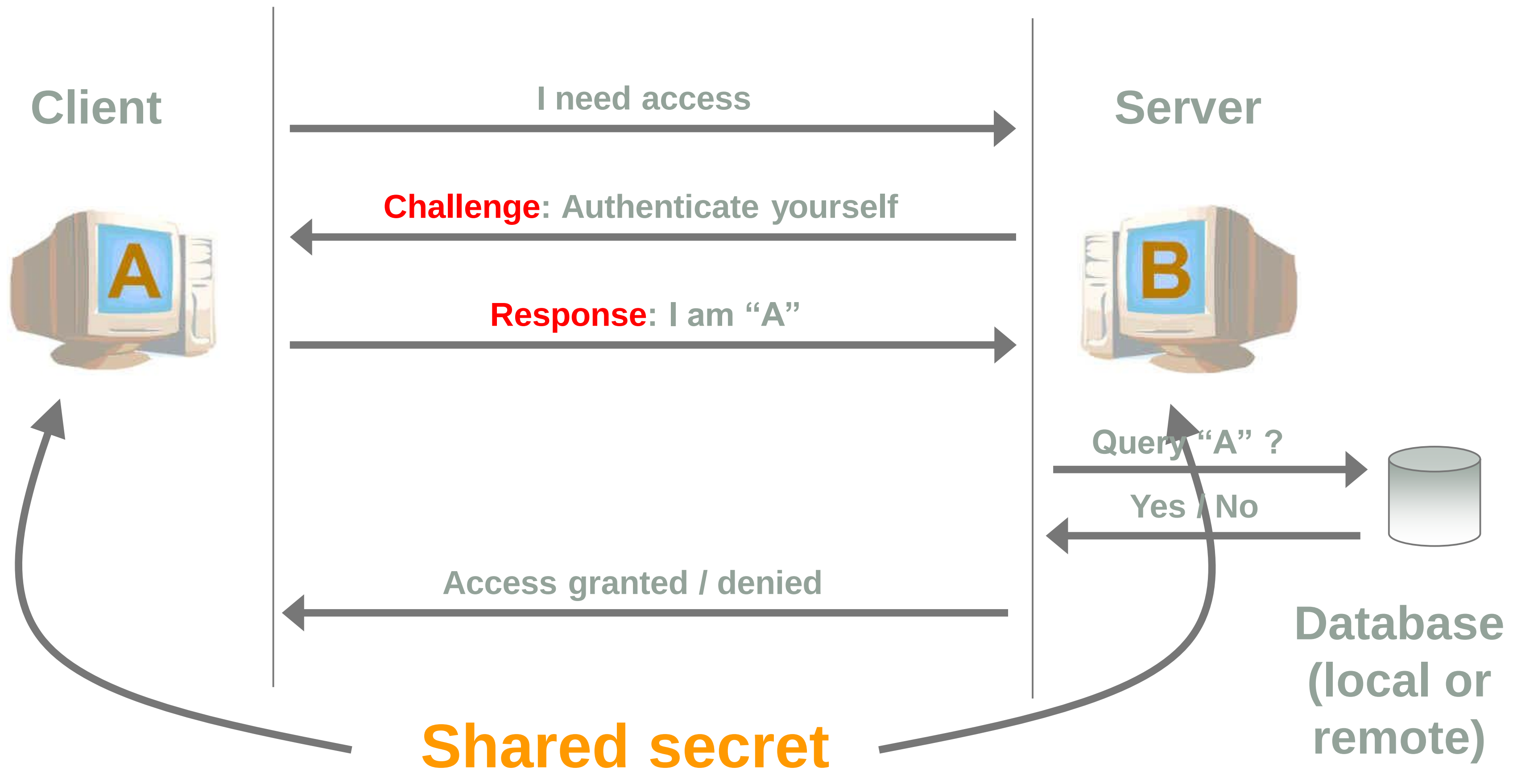
ACCESS

Authentication, Authorization, Accounting (AAA)

- **A**uthentication:
  - Critical to any security solution
  - Fundamental for access control (friend or foo)
- **A**uthorization:
  - Once authenticated, what resources are available for this user?
  - Database, files, services, hosts, etc.
  - Often (mistakenly) confused with Authentication
- **A**ccounting:
  - Logging of user activity
  - Can be used for:
    - Keeping a trace/log for security audits
    - Charging customers for resources usage (ISP)

Generic Authentication setup

Challenge-response: a common scheme for authentication



Authentication Factors

- User IS
  - Biometric data
- User HAS
  - Token
  - ID card
  - Cell Phone (SIM)
- User KNOWS
  - Password
  - Passphrase
  - PIN

Authentication is linked to having
Strong identities

Multi-factor Authentication

Best authentication systems are multi-factor

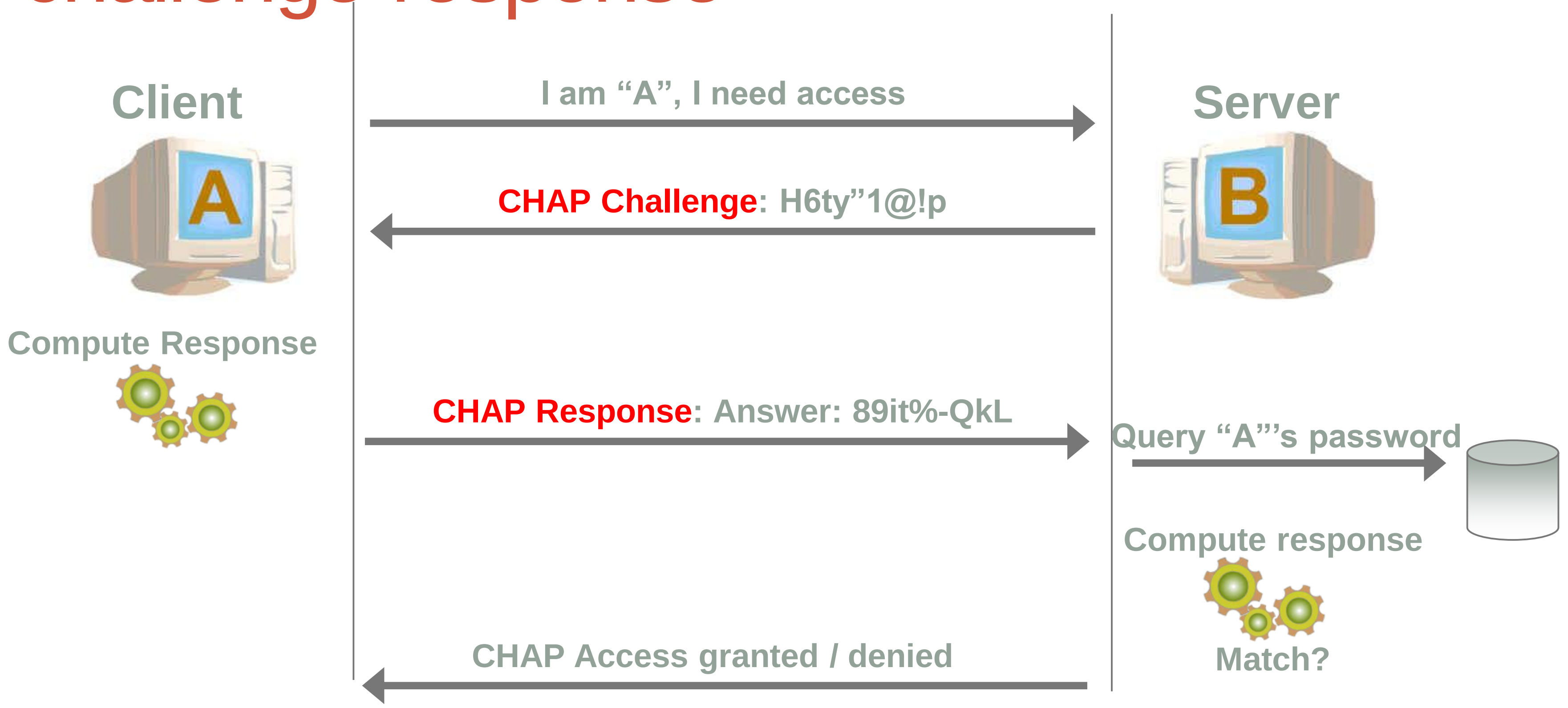
Popular:

- Two-factor authentication
  - Password + Token

High-end:

- Three-factor authentication
  - PIN + ID card + Biometric data

Password Authentication – CHAP challenge-response



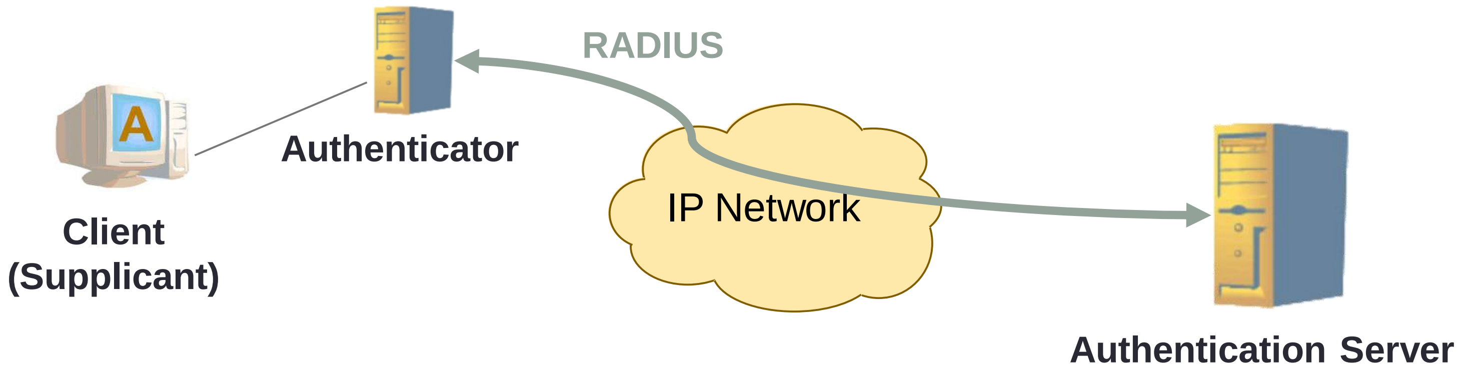
- Shared secret: password (could still be “QWERTY”)
- Challenge by B: randomly generated number
- Response by A: Hash of (Password + Random Challenge)
- Used by CHAP

Security of CHAP

- Cheating CHAP
  - Misusing the client:
  - Cheating CHAP (2002), Sebastian Krahmer
- Cryptographic properties of the hash algorithm
  - One-way,
  - 2nd pre-image resistant
- Hash size is important, why?

RADIUS
Remote Authentication Dial-In User Service

- Standardized in RFC 2865
- Three-party authentication
  1. **Client (supplicant),** request access to the network
  2. **Authenticato**r is the supplicant's first contact to the network, e.g.
     - Ethernet switch
     - Wi-Fi Access point
     - PPP end-point
  3. **Authentication Server** contains the user database and handles access request

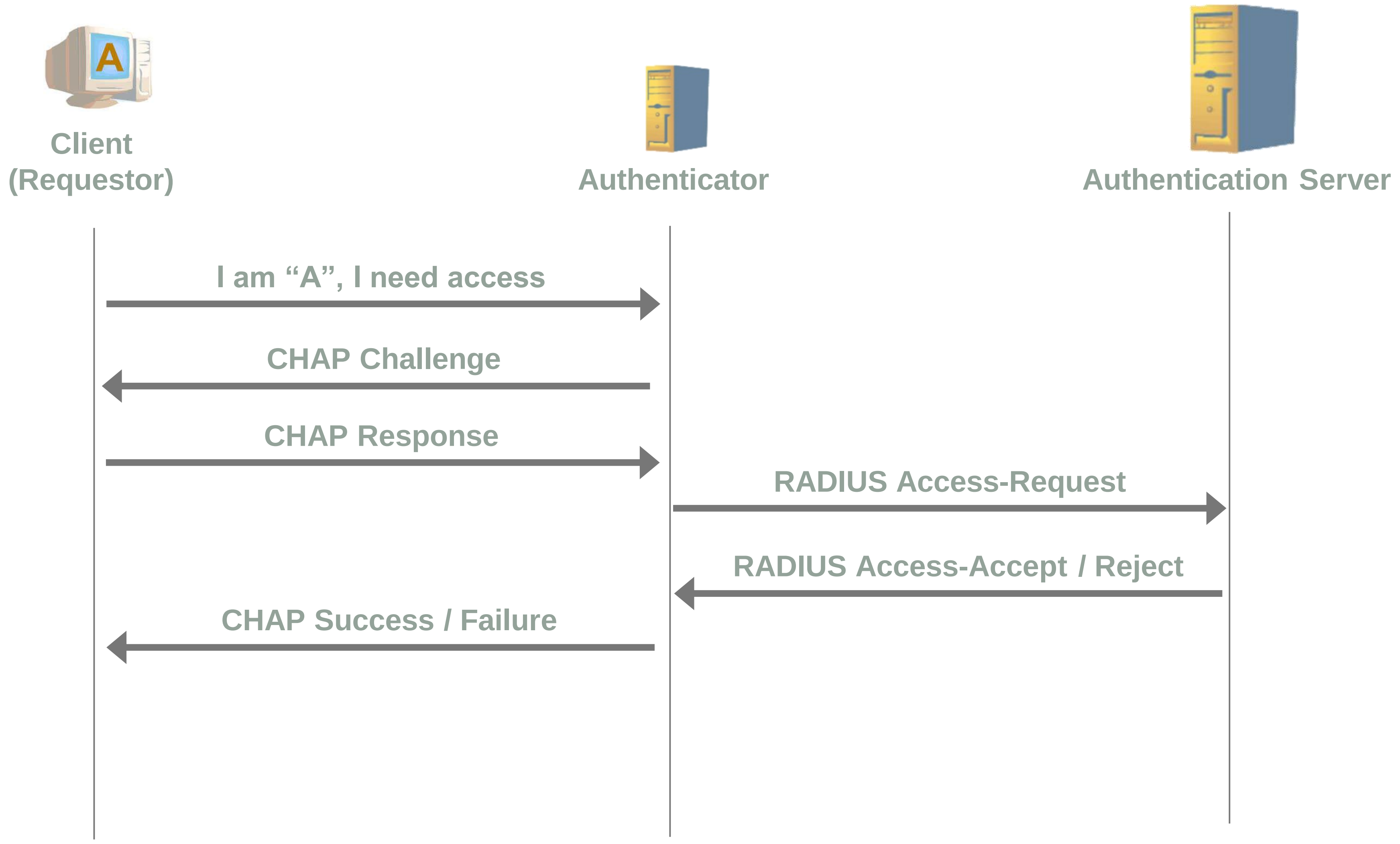


Important considerations

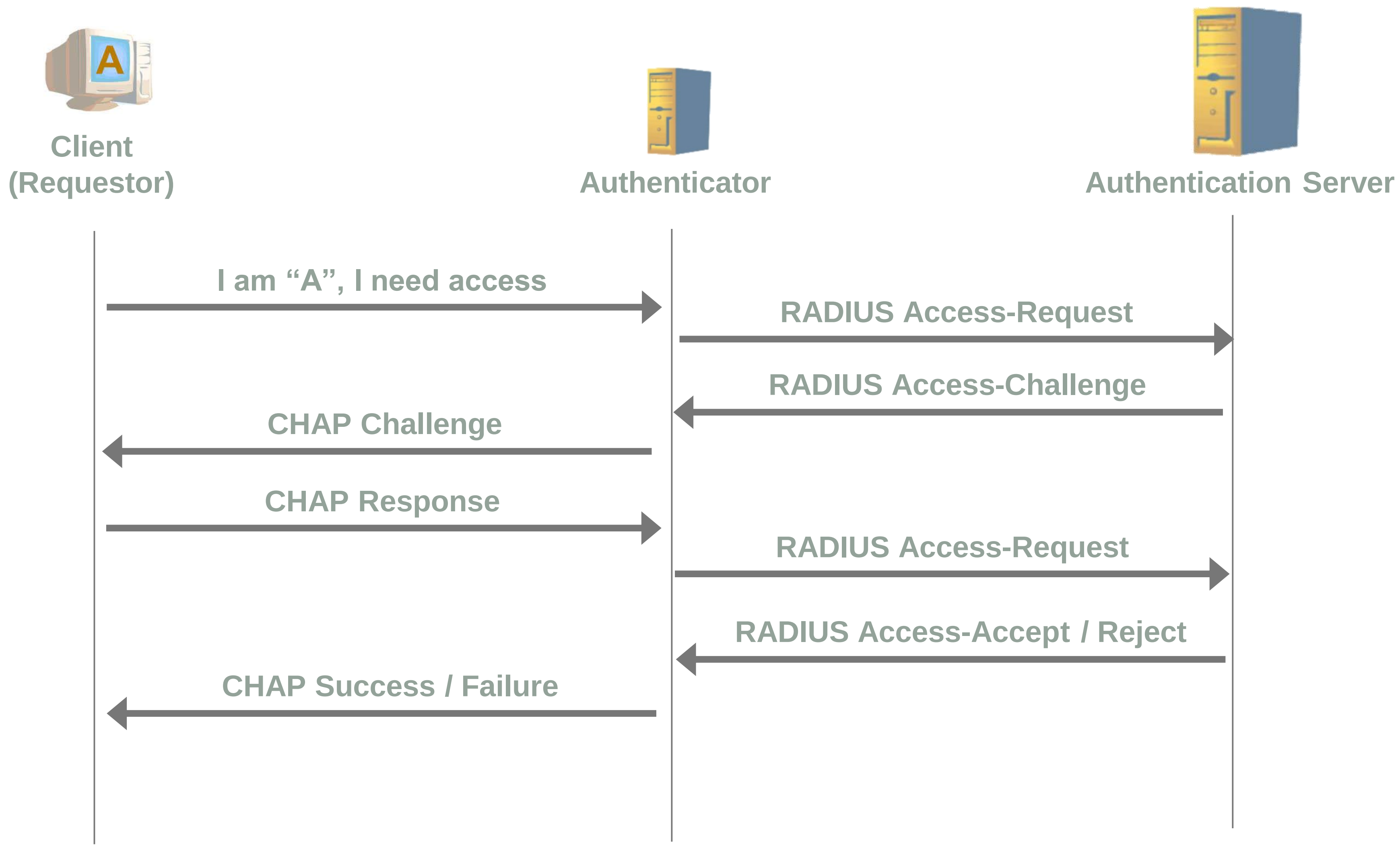
Beside the natural security issues (choice of algorithm, random generator, trust relations between entities, and storage of credentials)

- Scalability:
  - how many users and how often is authentication needed. E.g in Mobile systems we have millions of users.
- Access protection to authentication service

RADIUS Scenario 1 (with CHAP)



RADIUS Scenario 2



Notes on RADIUS Scenarios 1 and 2

- Why use one method or the other?
  - The Authenticator might be programmed to off-load the Authentication Server and will provide the CHAP Challenge itself
  - It might be better to have the Authentication Server provide challenges from a central location to assure uniqueness of challenges and prevent replay attacks
    - For example, when the Authenticator is a wireless Access Point
- In both scenarios, the Authenticator and Authentication Server share a secret key that is used to further encrypt the information and prevent replay attacks on data they exchange.

DIAMETER (successor of RADIUS)

- DIAMETER (is twice the RADIUS !)
- Standardized in RFC 3588
- Natural evolution of RADIUS to circumvent problems:
  - Better integration with upper-layer security protocols (e.g. TLS)
  - Better fail-over scenarios (for carrier environments for example)
  - Better audit support
  - Support for Mobile IP
    - This was a key determinant in choosing DIAMETER over RADIUS in 3G wireless networks

For more info see: Comparison between RADIUS and Diameter, A Hosi, HUT, Finland, 2003.

The evolution of PAP and CHAP: EAP

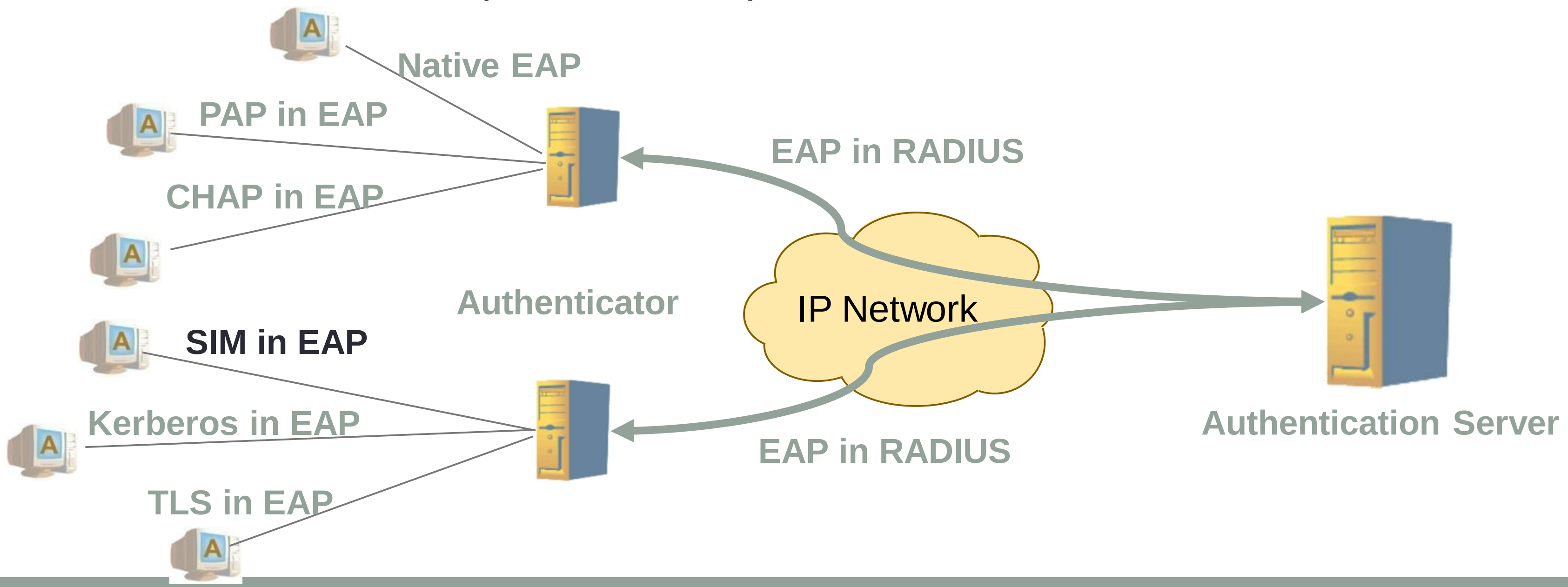
- Because network access may be requested by other means than the old CHAP, it was necessary to look for ways to accommodate flexibility

- A new protocol was created:

  Extensible Authentication Protocol (EAP)
  - Is a framework to support different authentication methods
  - has certain predefined auth methods

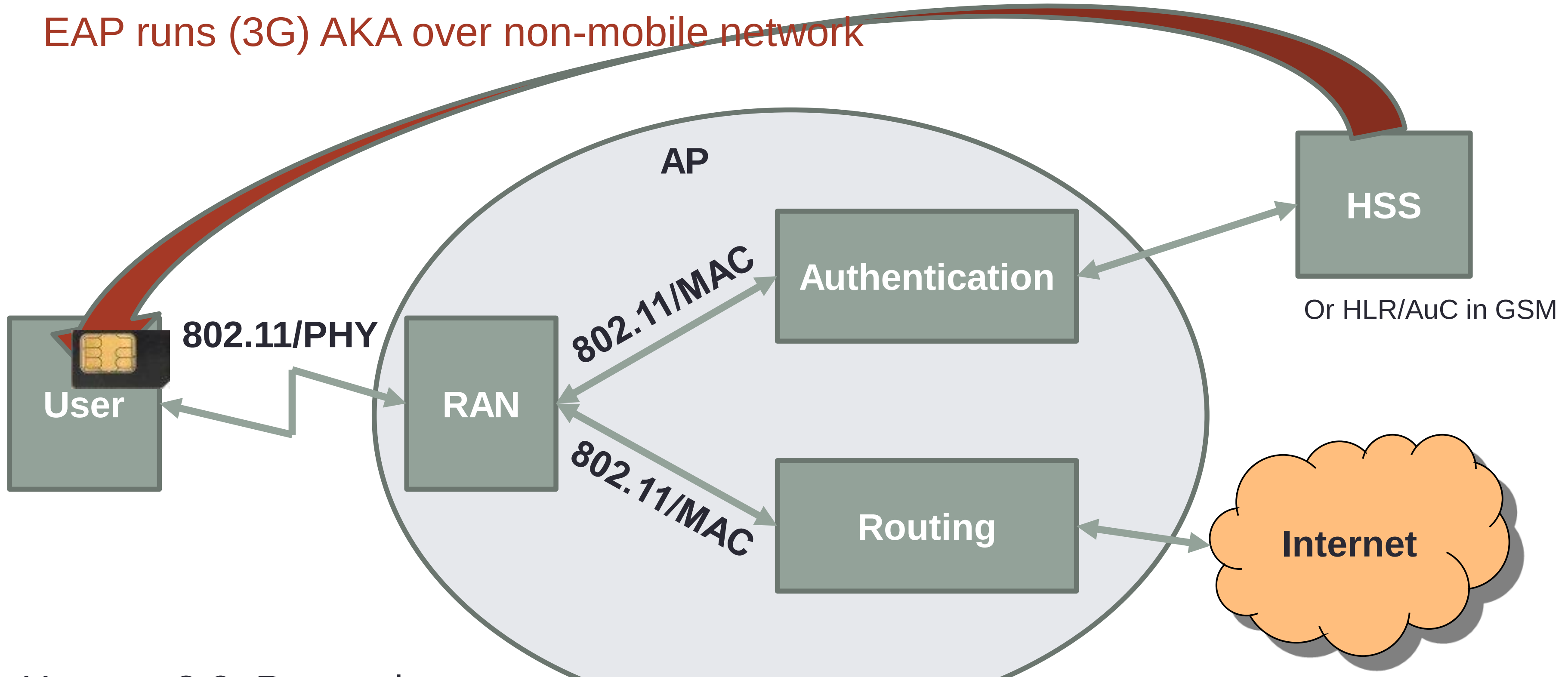
EAP Extensible Authentication Protocol

- Standardized in RFC 3748
- Designed for network access purposes
- Extensible means it supports existing and future authentication techniques:
  - PAP & CHAP
  - Token methods (covered later)
  - Upper Layer authentication such as TLS and Kerberos (covered later)



EAP-AKA: seamless access to WiFi

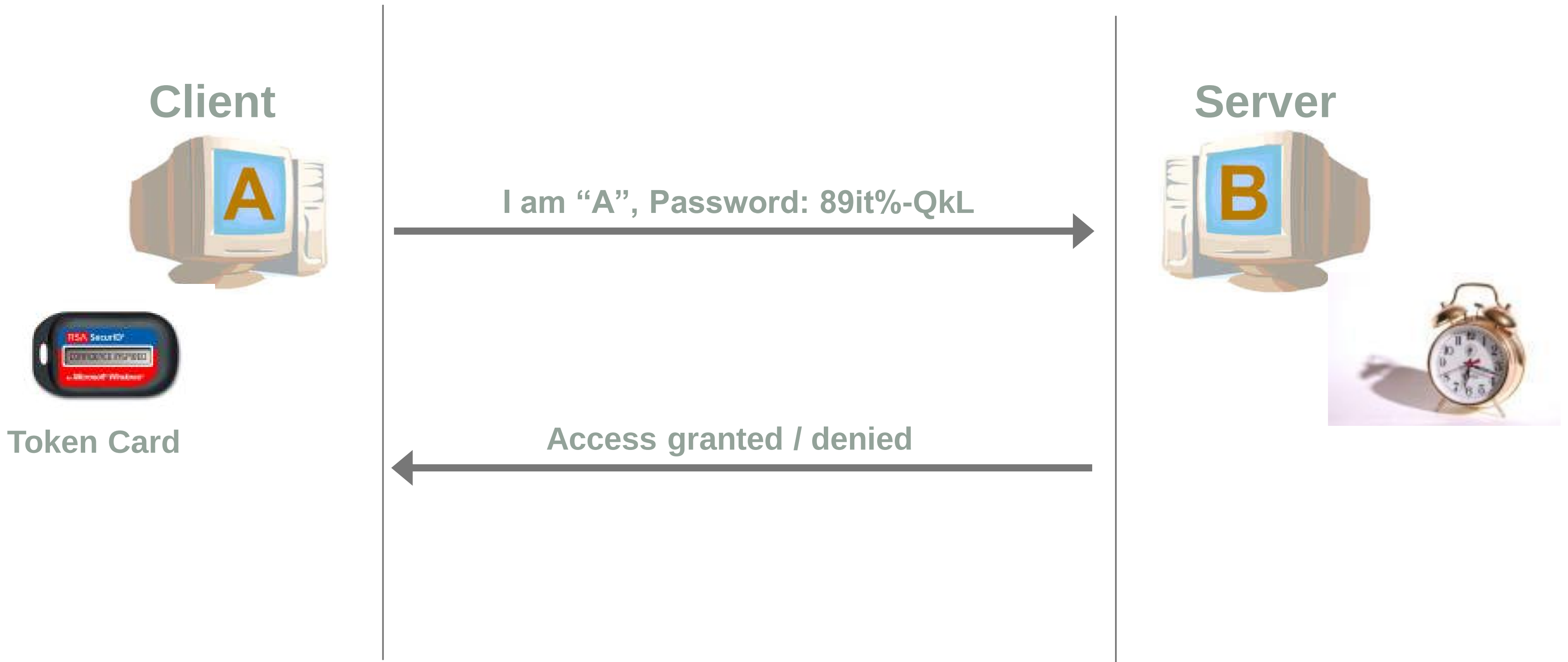
Network acces via WiFi for Mobile network offloading



Hotspot 2.0, Passpoint
(alternative AnyFi)

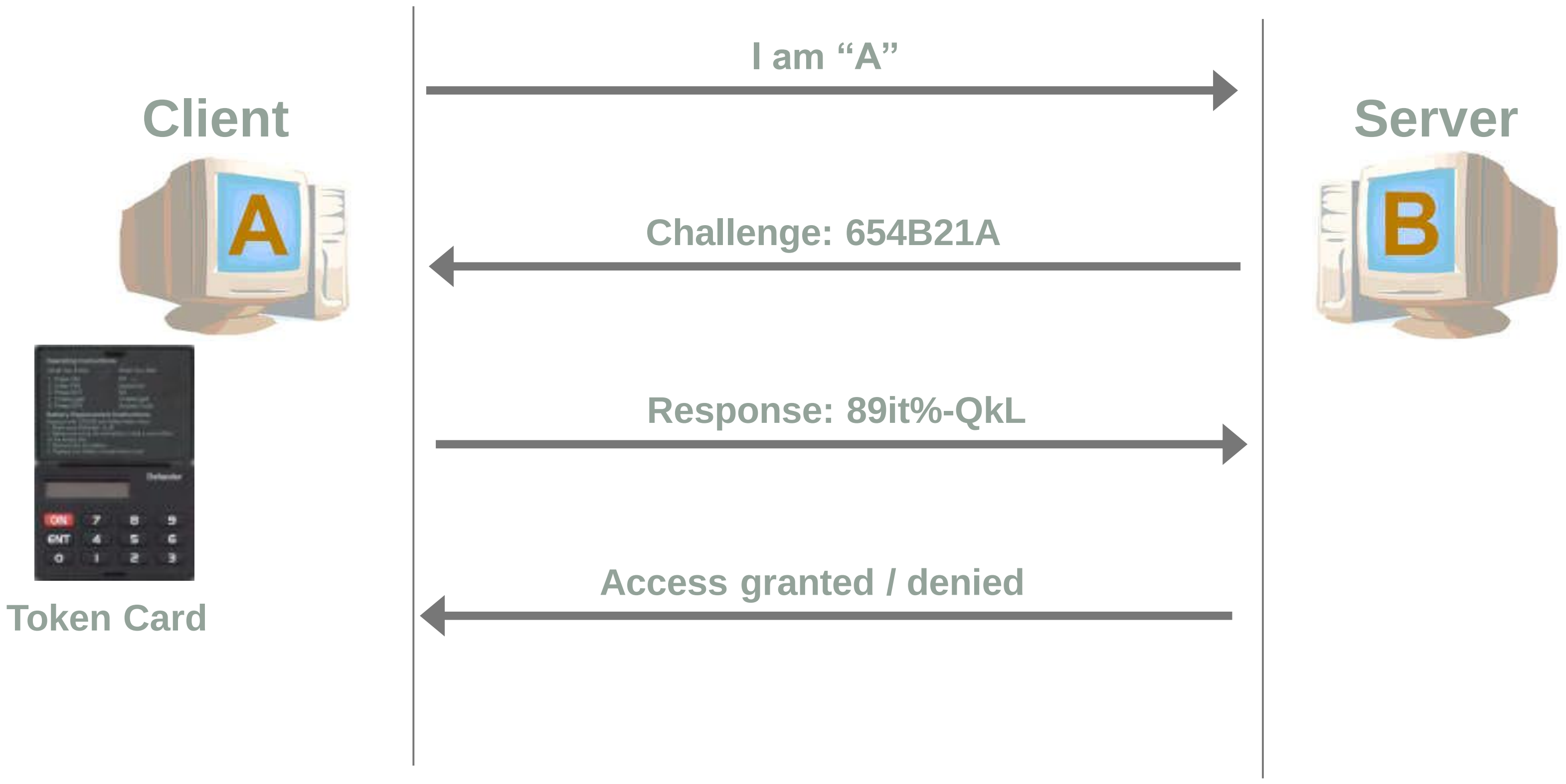
There is also EAP-SIM (GSM auth)

Token Authentication - synchronous



- Password is encrypted using a shared secret key between A and B, provided by the token
- The value on A's token card's screen changes periodically, in synch with B's internal clock

Token Authentication - asynchronous



- Password is encrypted using a shared secret key between A and B, provided by the token
- The response value on A's token card's screen is displayed when the user enters the challenge value

Single Sign-On and Federation

Single Sign-On and federated identity/access is not treated in this course. These solutions use treated in the Advanced Web Security course.

Single Sign-On (SSO) allows users to access multiple services with a single login.SSO is, unfortunately used ambiguously.

1. It can mean that the user only has to provide credentials a single time per session, and then gains access to multiple services without having to sign in again during that session. (this is what a call true SSO)
2. Or it is used to characterize a setup where the same credentials are used for multiple services; the user might have to login multiple times, but it's always the same credentials.

Federated Identity (FID) refers to a setup where Identity Management systems are connected together. In FID, a user's credentials are always stored by a so-called "identity provider". When the user logs into a service, instead of providing credentials to the service provider, the service provider trusts the identity provider to validate the credentials.

(think about using Facebook, or Google accounts for access to other services)

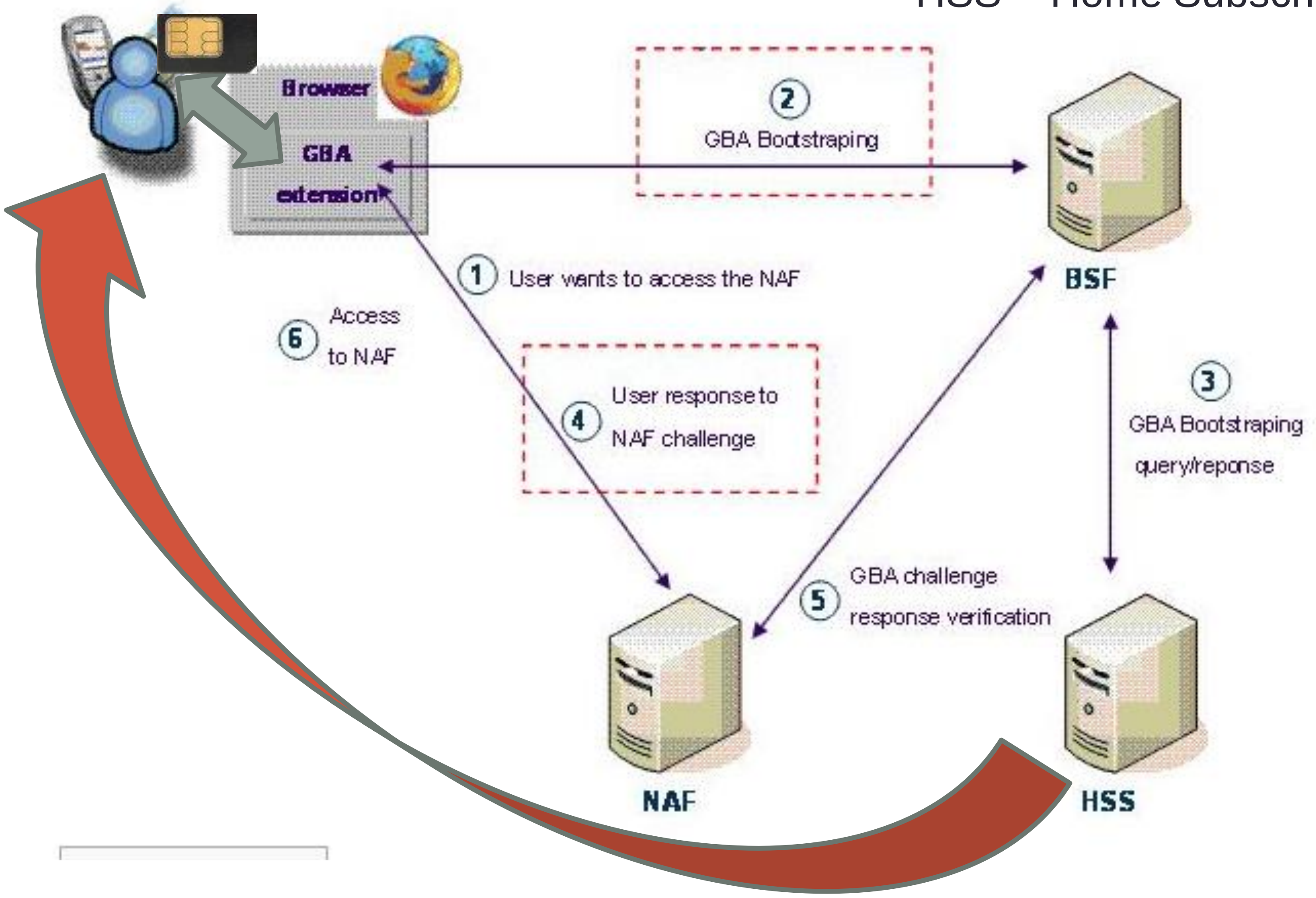
FID and SSO are different concepts but are very often used together.

Authentication in GSM/3G/LTE

- **Network Access** (Subscriber) authentication
  - Via SIM card: we come back to this later
- **Application authentication**
  - Generic Bootstrap Algorithm (GBA)
  - OpenID based on SIM

GBA get security keys from SIM at application level

NAF = Network Application Function
BSF = Bootstrap Server Function
HSS = Home Subscriber Server


Browser
GBA extension
2 GBA Bootstraping
BSF
1 User wants to access the NAF
6 Access to NAF
4 User response to NAF challenge
3 GBA Bootstraping query/reponse
5 GBA challenge response verification
NAF
HSS

GBA get security keys from SIM at application level

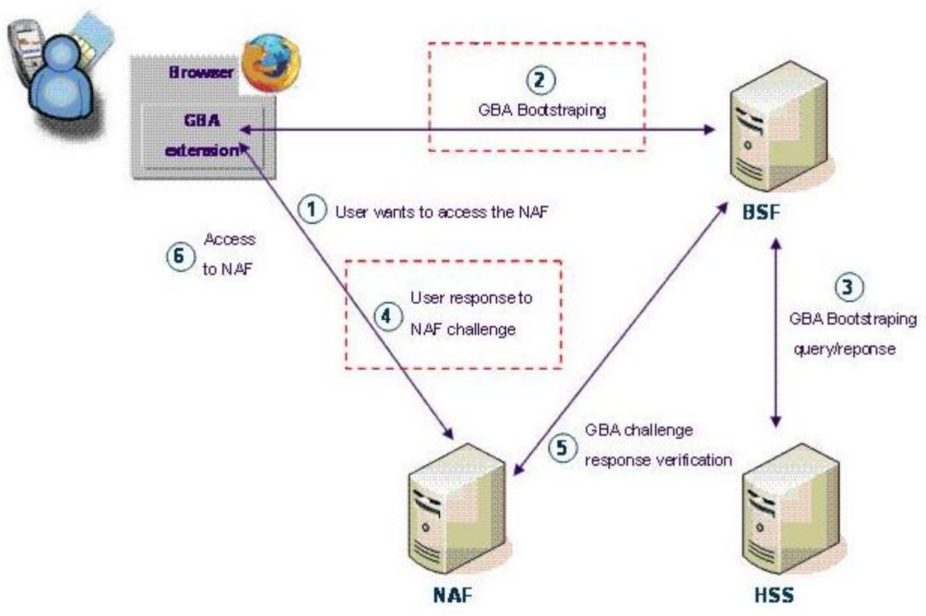
1 Ask access to NAF

2 Authenticate user

3 Get NAF key

4-5 NAF-user authentication

6 Access to NAF


Browser
GBA extension
2 GBA Bootstraping
BSF
1 User wants to access the NAF
6 Access to NAF
4 User response to NAF challenge
3 GBA Bootstraping query/reponse
5 GBA challenge response verification
NAF
HSS

NAF = Network Application Function
BSF = Bootstrap Server Function

Conclusion

- Authentication is a critical step in system security
  - Separates the legitimate users from the “bad guys”
    - If authentication is performed or bypassed by an attacker, the attacker will be considered by the system as a legitimate user.
      - SO THINK ABOUT THE CONSEQUENCES
  - Must be performed at the beginning of a transaction and may be subsequently repeated (at regular intervals)
    - but the latter is very seldomly done as it is very user unfriendly
  - Has mechanisms to prevent attacks:
    - Replay attacks
    - Password attacks (dictionary, brute-force)

Conclusion (cont’d)

- As security administrators, NEVER rely on authentication alone
  - Should be part of a greater security policy: defense in depth
  - Use authorization and intrusion detection

AND FOREMOST

- We need “secure identities”, that is secure credentials-identifier pairs that can securely associated with users (not necessary humans only).

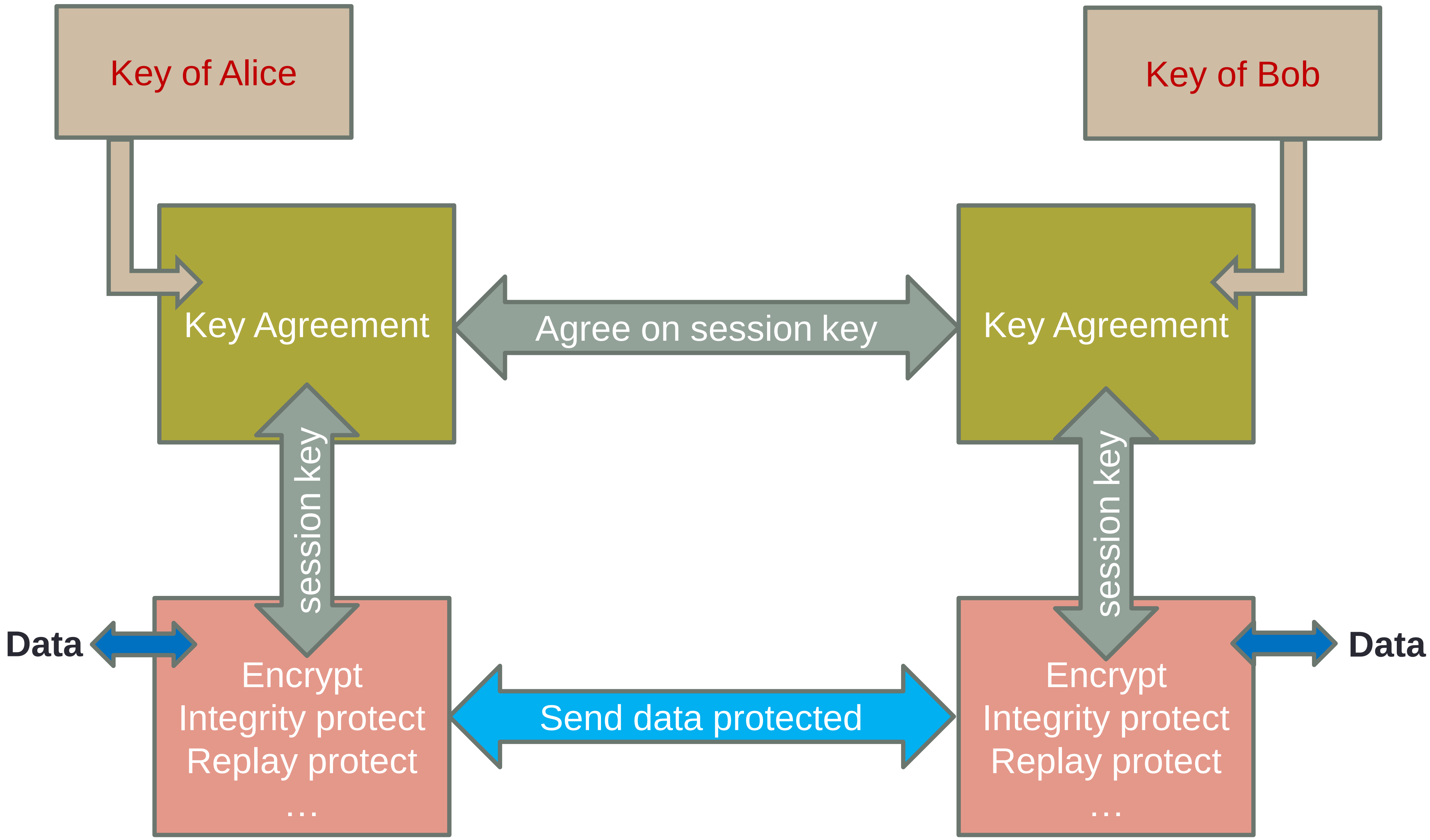
We come back later to secure identities.

PART III
SECURE CONNECTIONS

Overview

- The general picture
- Reflections on
  - TLS, DTLS
  - IPsec
- Opportunistic encryption
- Object security
- LTE Security
  - Better security for handover

The general picture



Advantage of the split

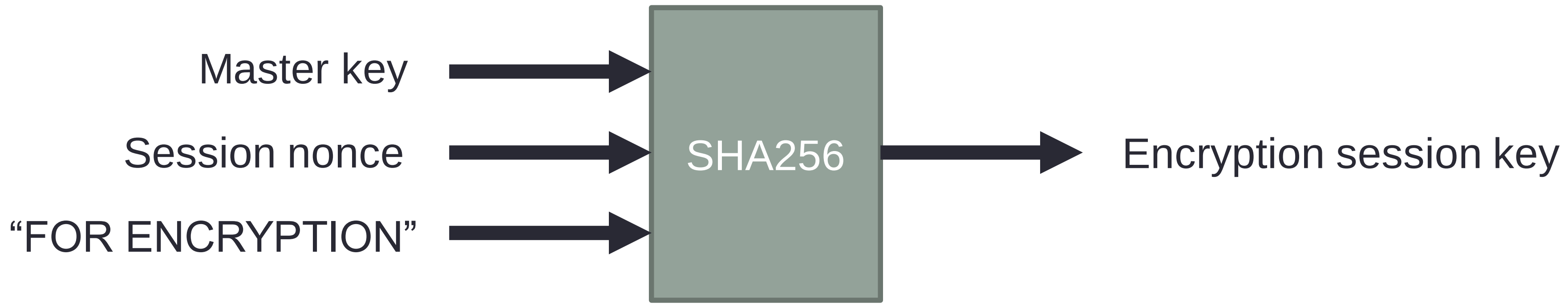
- Although the explicit split may result in a more complex implementation it has several advantages

- Encryption should not be used too long with same key.
- Multiple application and multiple threading easier
  - Encryption must run as fast as possible and should handle data from any application
  - Key agreement may have demands on application and vice versa.
- Modular: easier to develop and maintain and replace or expand when algorithms become obsolete.

Key derivations

- Protocols use key derivation functions which derive other keys such as session keys or separate (why?) keys for encryption and keys for integrity protection.

- A key derivation is likely realized using a hash or MAC function using the (master) key and, often, specific random or nonce data that is mixed-in the computation, e.g.



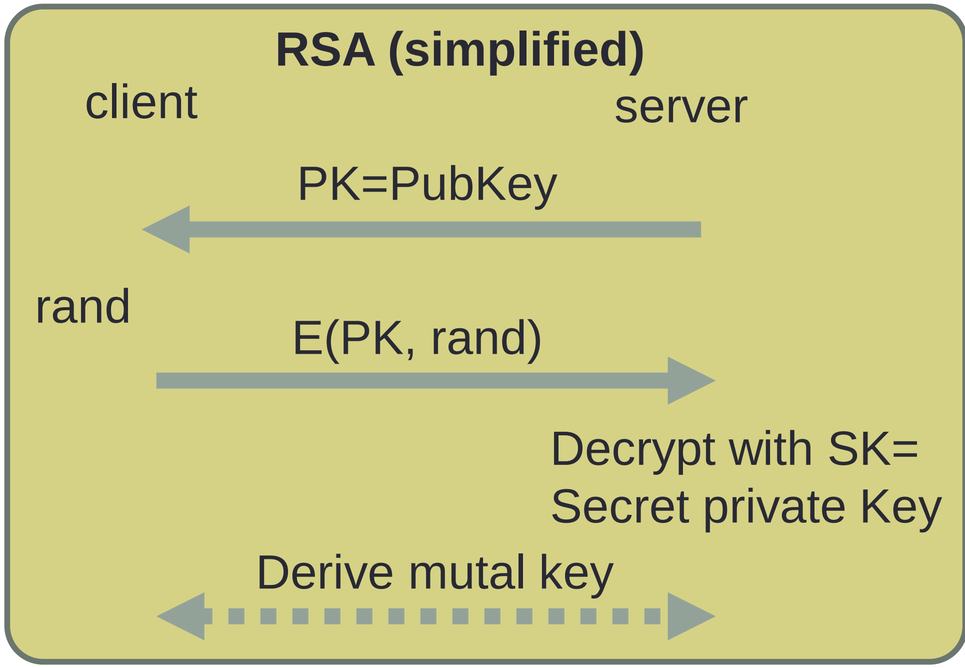
Reflections on: TLS/DTLS

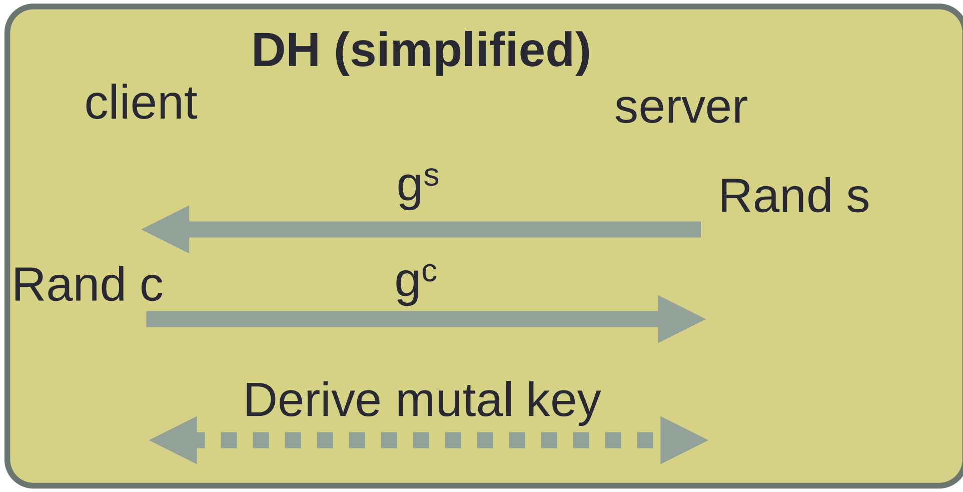
- DTLS is like TLS but runs over UDP instead of TCP
- Both TLS and DTLS
  - have built-in key agreement
    - But one can run them also in Pre-Shared Key (PSK) mode
  - Sender and receiver must maintain a state for the session
  - Encrypted data cannot be cached for transport later

- Which protection to use?
  - Choice of algos for key agreement
  - Choice of encryption algorithm
  - Choice of data integrity protection algorithm

Choice of key agreement

- Security (strength)
- Complexity, e.g. DH vs RSA
- PQC aspects?
- Protocol
  - Easier to get at the mutual session in RSA than in DH. why?

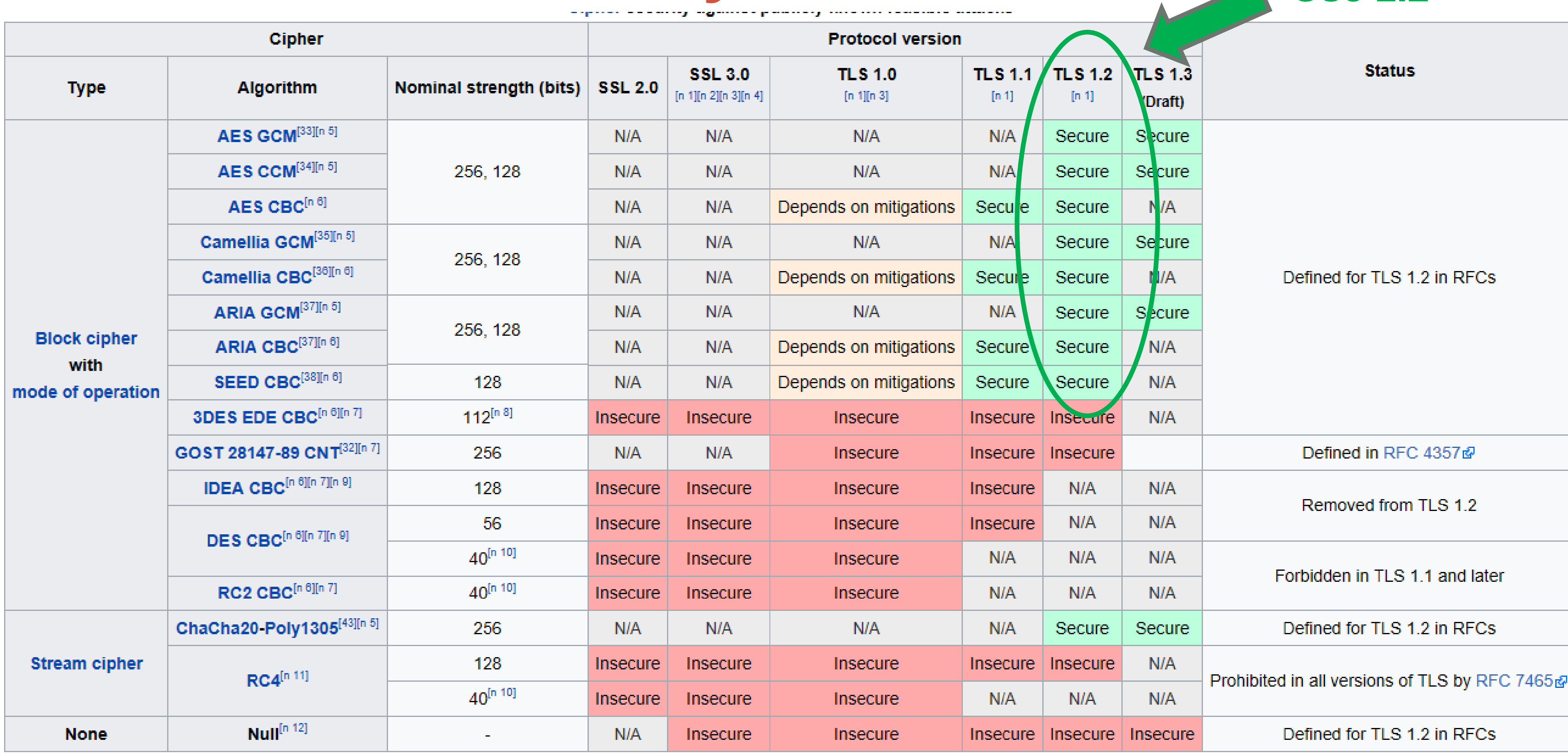




Recall that complexity of a^x mod n is for random a in [1..n-1] (using schoolbook technique) C log(x) (log(n))^2 , for some constant C

SSL/TLS Security Flaws -status

Use 1.2



Cipher			Protocol version						Status
Type	Algorithm	Nominal strength (bits)	SSL 2.0	SSL 3.0 [n 1][n 2][n 3][n 4]	TLS 1.0 [n 1][n 3]	TLS 1.1 [n 1]	TLS 1.2 [n 1]	TLS 1.3 (Draft)	
Block cipher with mode of operation	AES GCM[33][n 5]	256, 128	N/A	N/A	N/A	N/A	Secure	Secure	Defined for TLS 1.2 in RFCs
	AES CCM[34][n 5]		N/A	N/A	N/A	N/A	Secure	Secure	
	AES CBC[n 6]		N/A	N/A	Depends on mitigations	Secure	Secure	N/A	
	Camellia GCM[35][n 5]	256, 128	N/A	N/A	N/A	N/A	Secure	Secure	
	Camellia CBC[36][n 6]		N/A	N/A	Depends on mitigations	Secure	Secure	N/A	
	ARIA GCM[37][n 5]	256, 128	N/A	N/A	N/A	N/A	Secure	Secure	
	ARIA CBC[37][n 6]		N/A	N/A	Depends on mitigations	Secure	Secure	N/A	
	SEED CBC[38][n 6]	128	N/A	N/A	Depends on mitigations	Secure	Secure	N/A	
	3DES EDE CBC[n 6][n 7]	112[n 8]	Insecure	Insecure	Insecure	Insecure	Insecure	N/A	
	GOST 28147-89 CNT[32][n 7]	256	N/A	N/A	Insecure	Insecure	Insecure		Defined in RFC 4357
	IDEA CBC[n 6][n 7][n 9]	128	Insecure	Insecure	Insecure	Insecure	N/A	N/A	Removed from TLS 1.2
	DES CBC[n 6][n 7][n 9]	56	Insecure	Insecure	Insecure	Insecure	N/A	N/A	
		40[n 10]	Insecure	Insecure	Insecure	N/A	N/A	N/A	Forbidden in TLS 1.1 and later
	RC2 CBC[n 6][n 7]	40[n 10]	Insecure	Insecure	Insecure	N/A	N/A	N/A	
Stream cipher	ChaCha20-Poly1305[43][n 5]	256	N/A	N/A	N/A	N/A	Secure	Secure	Defined for TLS 1.2 in RFCs
	RC4[n 11]	128	Insecure	Insecure	Insecure	Insecure	Insecure	N/A	Prohibited in all versions of TLS by RFC 7465
		40[n 10]	Insecure	Insecure	Insecure	N/A	N/A	N/A	
None	Null[n 12]	-	N/A	Insecure	Insecure	Insecure	Insecure	Insecure	Defined for TLS 1.2 in RFCs

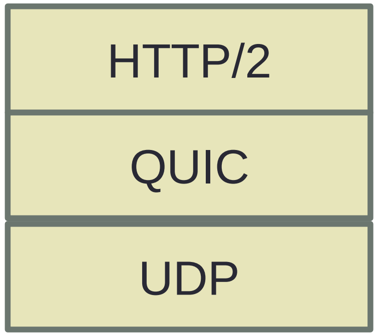
Source: Wikipedia 2017

Mandatory reading:

Cryptography in an all encrypted world , Ericsson Review, Dec 2015

Improvements

- TLS 1.3:
  - better – stronger and faster algorithms
  - Combine encryption and MAC in one algorithm
  - Drop support of compression
  - Less round trips
  - Removal of algorithms, e.g. MD5 and support for backward compatibility with SSL
- Google QUIC (Quick UDP Internet Connections)
  - Zero round trips!
  - Provides multiplexed in-order reliable stream transport (especially HTTP) over UDP
  - Protocol is pushed into application space (unlike TCP which is handled in kernel)



QUIC How to get zero round trip?

Speculate that the server's public key is unchanged since last contact

- Propose session encryption key in first packet

Include GET request(s) immediately after

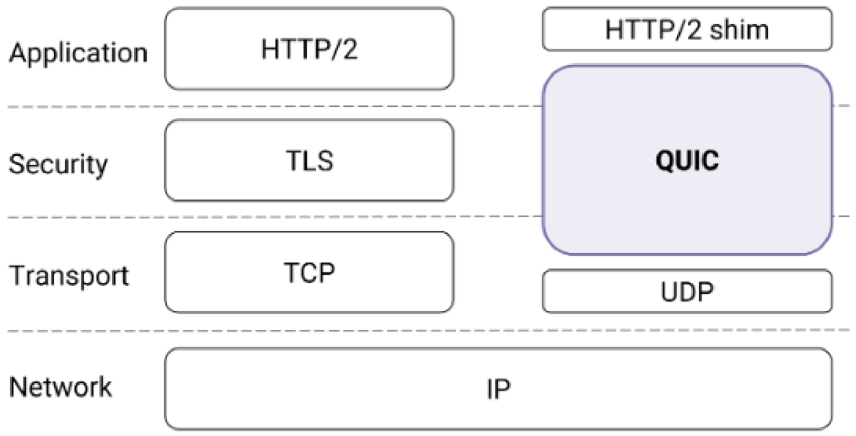


Figure 1: QUIC in the traditional HTTPS stack.

The QUIC Transport Protocol:
Design and Internet-Scale Deployment, Adam Langley, at all, SIGCOMM '17, August 21-25, 2017, Los Angeles, CA, US

IPsec

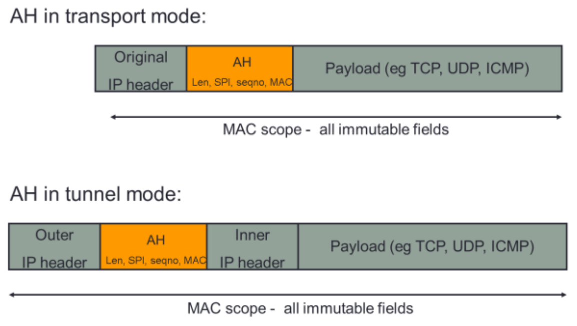
- Recall of some facts
- IKE v2
- NAT traversal

IPSec: Recall Basic Features

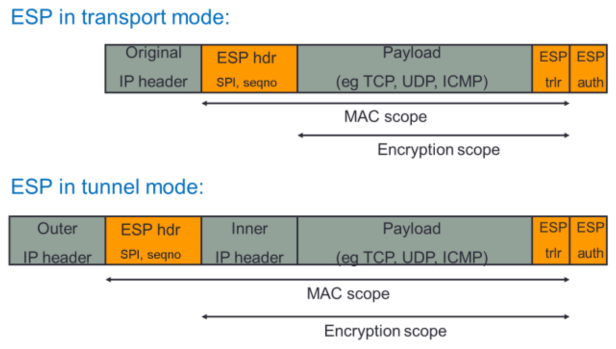
- IPSec provides two basic modes of use:
  - "transport" mode: for IPSec-aware hosts as endpoints.
  - "tunnel" mode: for IPSec-unaware hosts, established by intermediate gateways or host OS.

- IPSec provides authentication and/or confidentiality services for data with two protocols

AH protocol

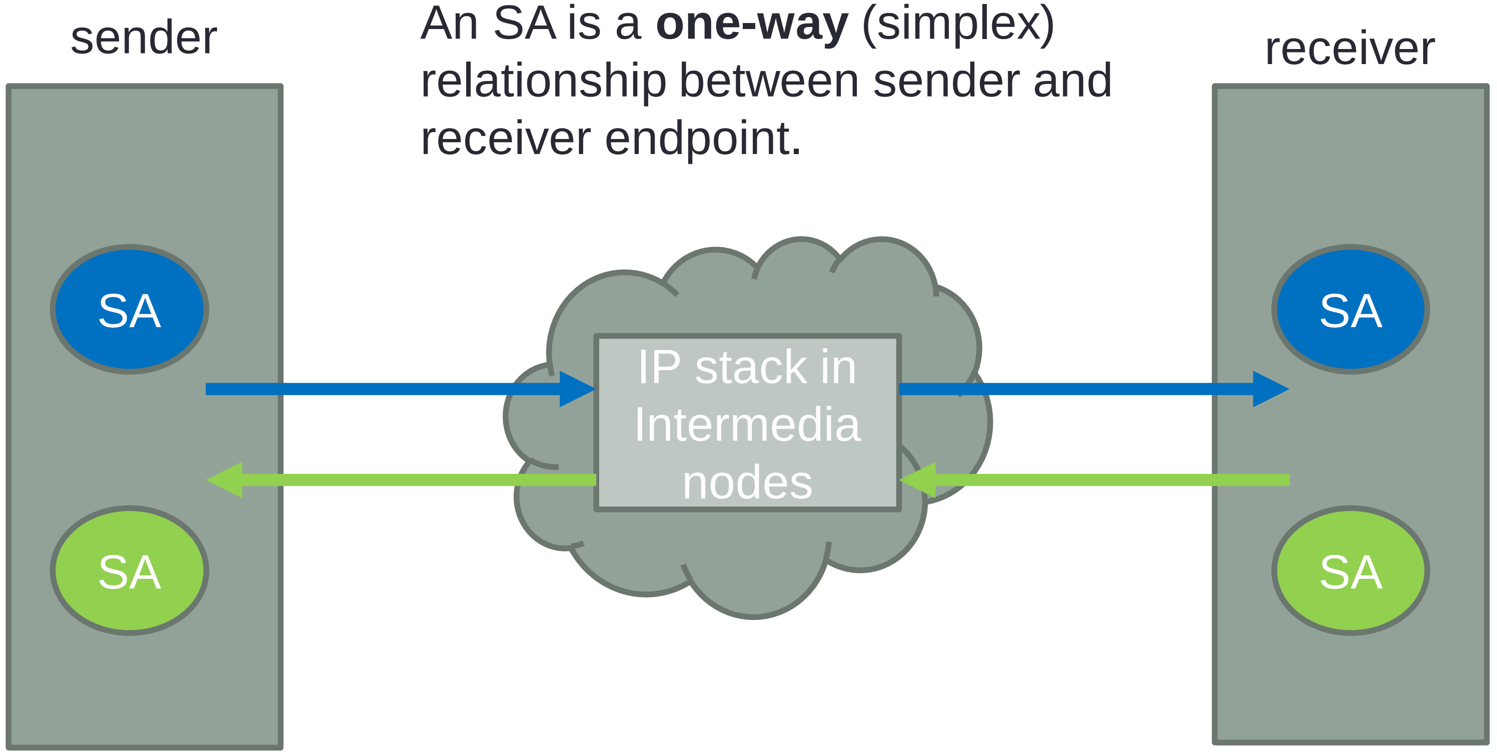


ESP protocol



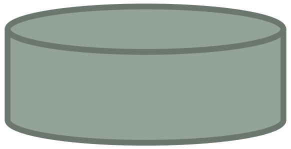
IPsec - endpoints

An SA is a **one-way** (simplex) relationship between sender and receiver endpoint.



IPSec Security Associations (SAs)

- An SA is a **one-way** (simplex) relationship between sender and receiver.
  - Specifies cryptographic processing to be applied to *this* datagram from *this* sender to *this* receiver.
- SAs are held in a SA database (SADB)
  - list of active SAs



- Each SA is identified by unique SPI (32 bit value carried in AH and ESP headers).
  - Allows recipient to determine how to process received datagrams.
- Each SA contains:
  - Sequence number counter and anti-replay window.
  - AH/ESP info: algorithms, IVs, keys, key lifetimes.
  - SA lifetime.
  - Protocol mode: tunnel or transport.
  - …

IPSec Key Management

- **IPSec is a heavy consumer of symmetric keys:**
  - One key for each SA.
  - Different SAs for:
    - {ESP,AH} x {tunnel,transport} x {sender, receiver}.

Where do these SAs and keys come from?

Two options

- **Manual keying.**
  - Fine for small number of nodes but hopeless for reasonably sized networks of IPSec-aware hosts; requires manual re-keying.

- **IKE: Internet Key Exchange**, **IKEv1** and IKEv2 (RFC 4306).
  - RFC documentation hard to follow.
  - IKE is a specific adaptation of more general protocols ("Oakley" and "ISAKMP").
  - Protocols have many options and parameters.

IKE Phases

IKE operates in two phases.

- **Phase 1** provides mutual authentication and establishes session key

  Set up an SA and secure channel to carry further SA negotiation, as well as error and management traffic.
  - Bi-directional.
  - Heavy-duty entity authentication and key exchange.
  - Establishes ISAKMP channel (IPSec key management protocol) – a secure channel for use in Phase 2.

- **Phase 2** allows for multiple security associations for same Phase 1 pair. SAs for general use are negotiated.
  - Fast negotiation takes place over Phase 1 secure channel.
  - Many Phase 2 runs allowed for each run of Phase 1.
  - Multiple SAs can be negotiated per run.

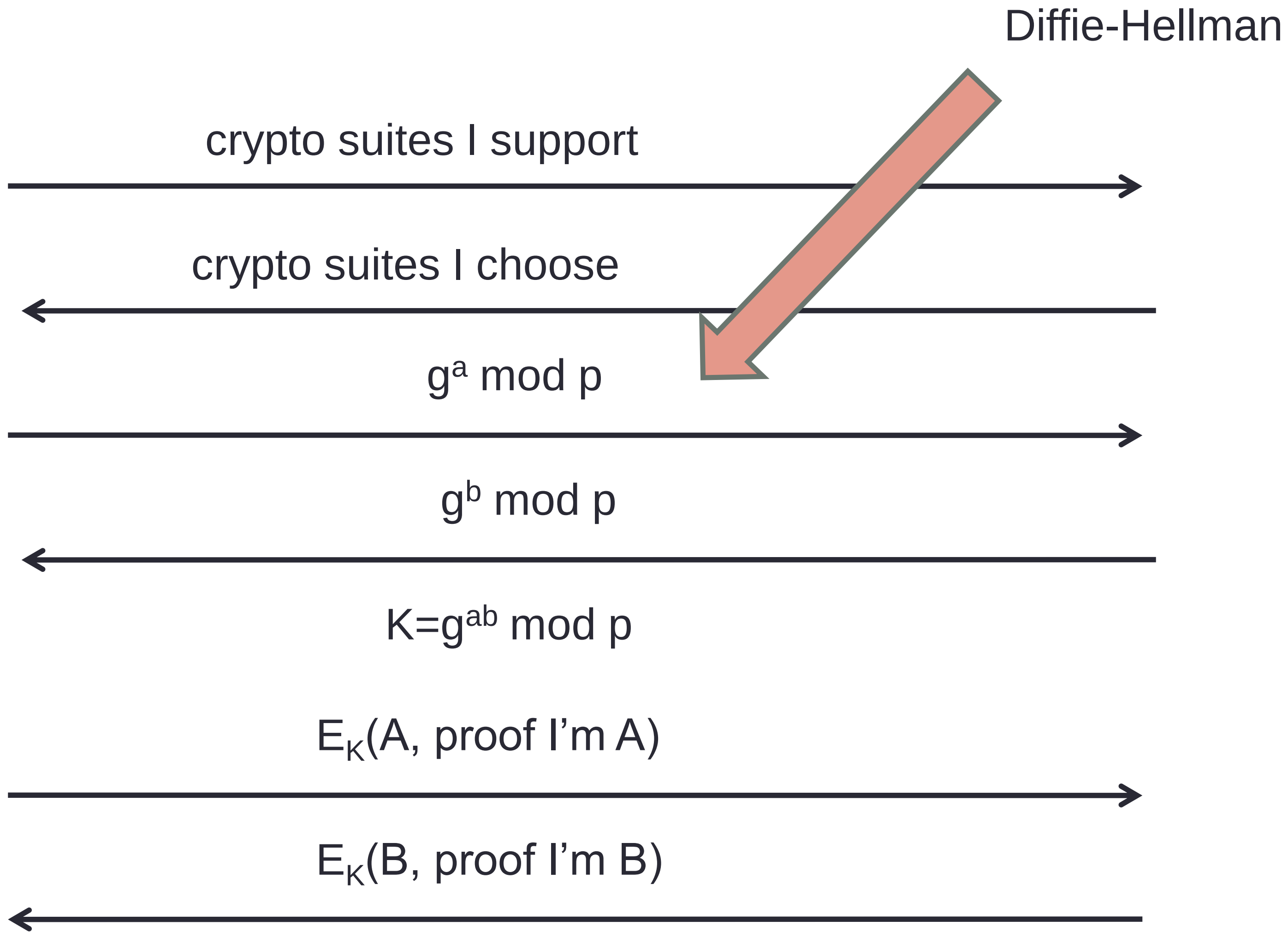
Phase 1 IKE

Two possible modes:

- **“Main mode”:** slow (6 messages), more cautious, hides details of credentials used and allows perfect forward secrecy -independence of short-term keys.

- **“Aggressive mode”:** less negotiation, only 4 messages, more information disclosed.

Phase 1 IKE Main Mode



IPSEC. IKE and NAT traversal

Parts of this material has been compiled from various open sources

IPsec and NAT: INTRODUCTION

NAT:

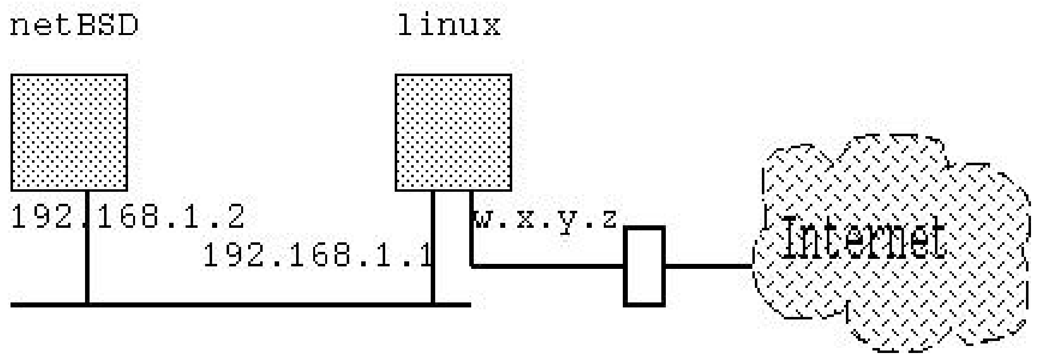
- NAT is router function that provides the network address translation between private and public IPv4 addresses.
  - IPv4 address space is limited

- Implementations: Static and dynamic

THUS:

NAT changes the source IP address of the packet.

Network Address Translation

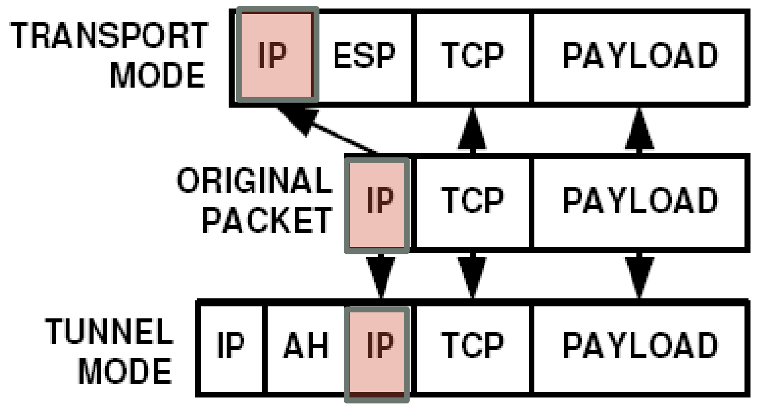


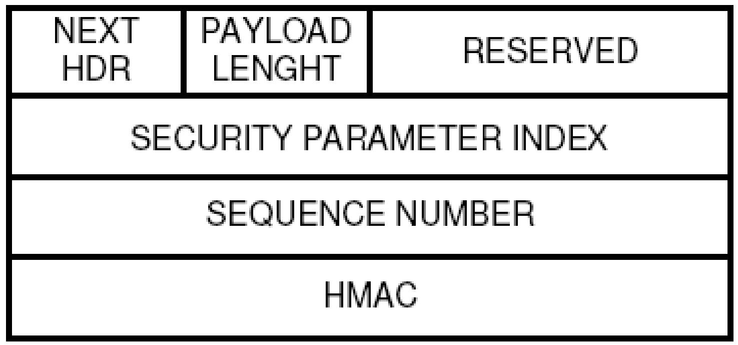
- Basic NAT
  - 192.168.1.2 → w.x.y.z

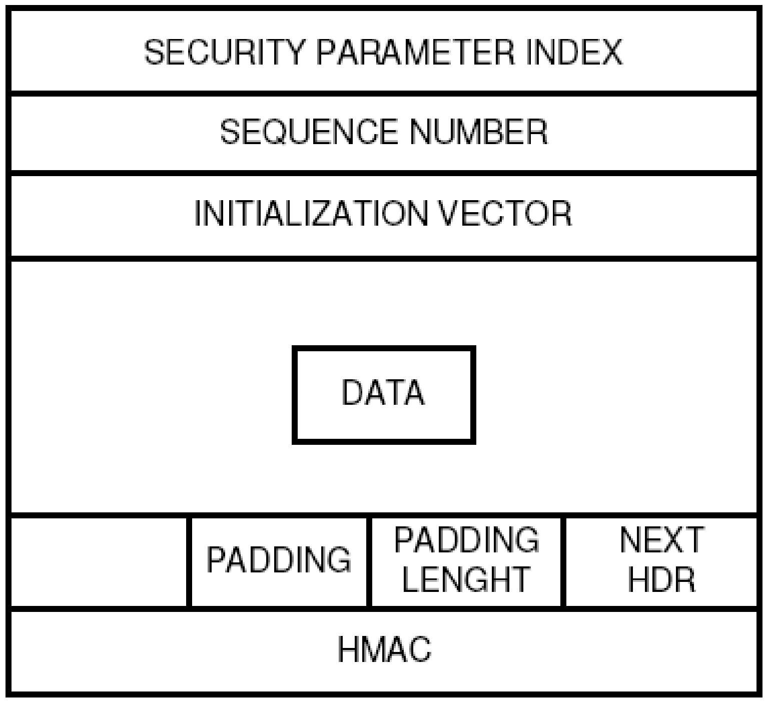
- NAPT (the most common form of NAT)
  - 192.168.1.2, Source Port A →w.x.y.z, Source Port B

address & port

IPSec processing and packet formats







IPSEC- NAT Incompatibility

- AH header incorporates the IP source and destination address for integrity check.
- TCP/UDP checksums.
- IKE-NAT Incompatibilities.

NAT-T: UDP encapsulation of IPsec ESP packets

- ESP: Only payload is encrypted

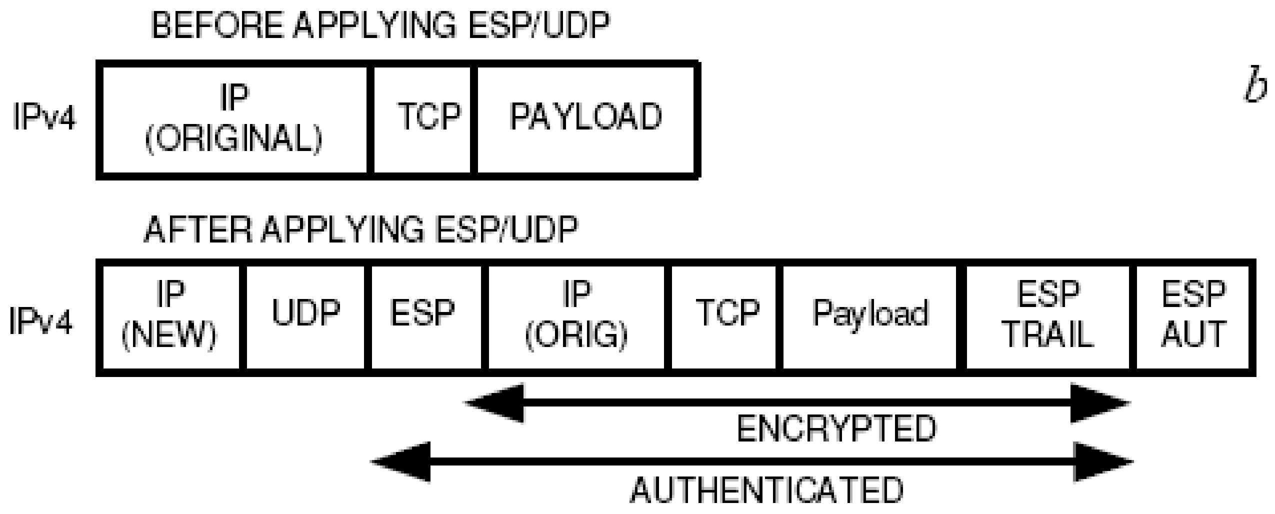
⇒ NAT-T adds a UDP header that encapsulates the ESP header.

Functionality: (during initial IPSec negotiation)

1. If peers have NAT-T capability
2. NAT router in the middle of the path between the peers

⇒ Otherwise normal IPsec operations

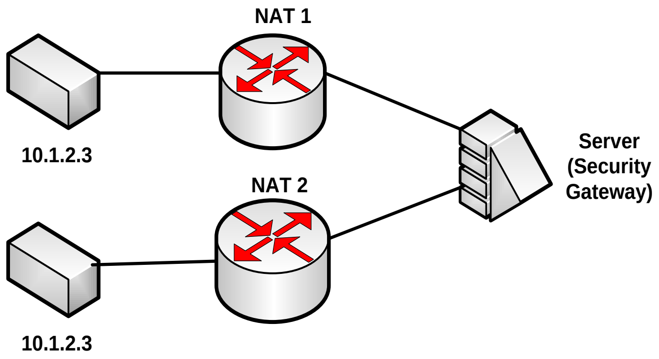
ENCAPSULATION (ESP: recall)



NAT-T PROBLEMS

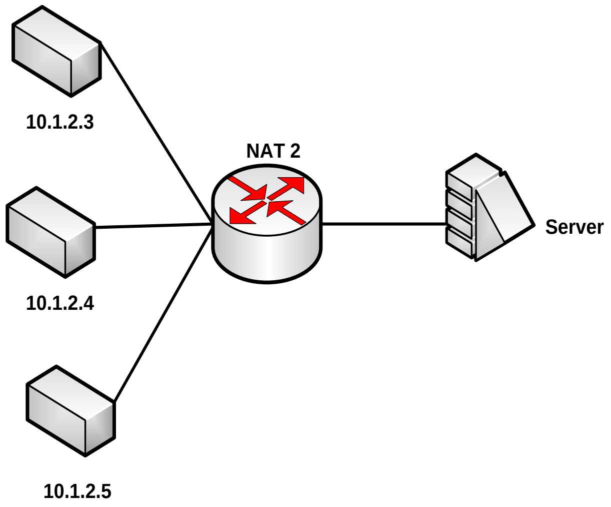
- **Tunnel mode conflict**

  Remote peers may negotiate entries that overlap when tunnel mode is used.



- **Transport mode conflict**

  May occur when two peers behind NAT routers are in communication with same server. Server may get confused which SA is belonging to which client.

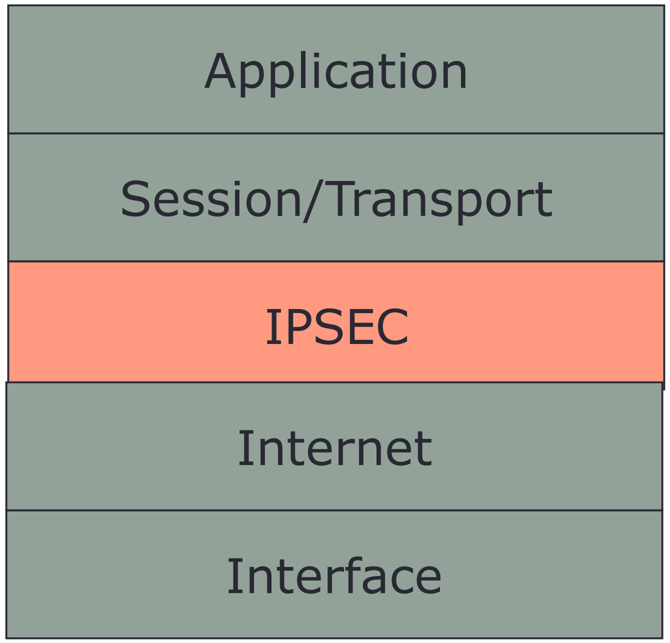


EXTRA BACKGROUND MATERIAL

For convenient reading some slides are repeated here

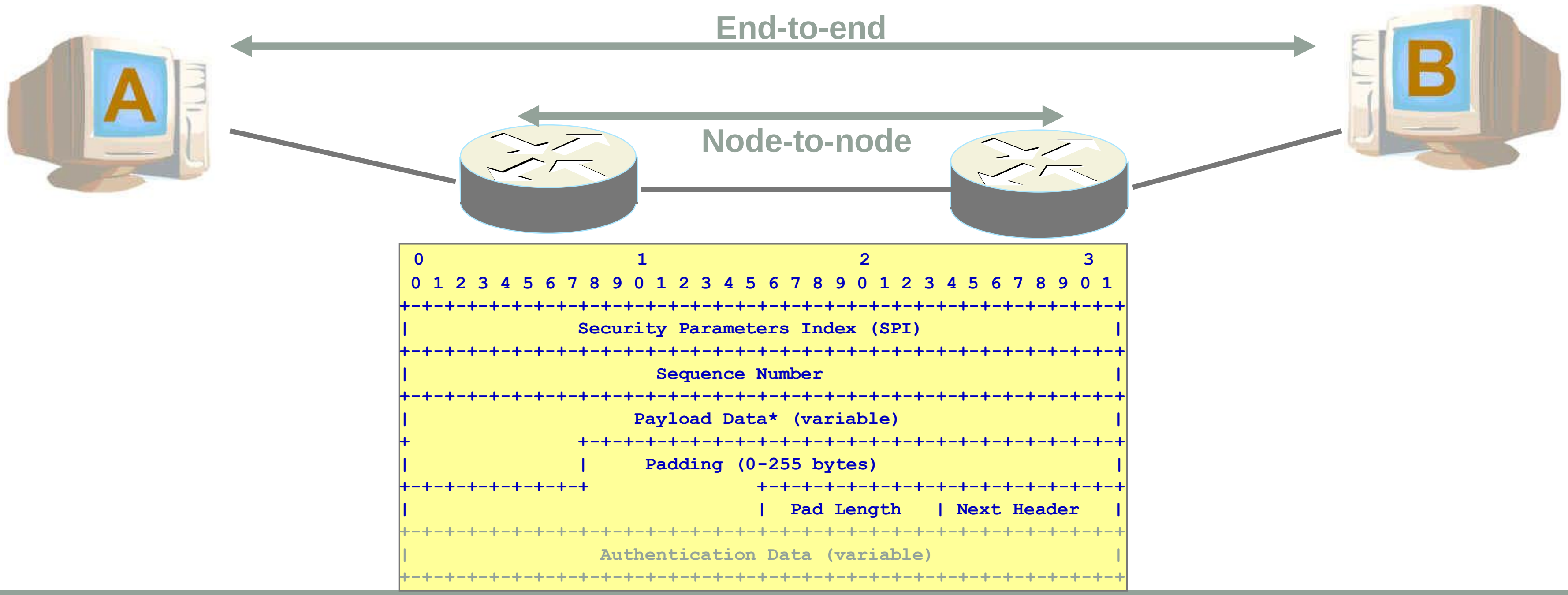
IPSEC

- transparent protection



IPSec/IKE – (secure tunnel/key agreement)

- RFC 2401 and 2406
- IPSec works at layer 3 of the TCP/IP stack model to create secure tunnels
- Each encrypted packet in the tunnel contains authentication data to prove its origin



```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Security Parameters Index (SPI)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Sequence Number                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Payload Data* (variable)                   |
+               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               |     Padding (0-255 bytes)                     |
+-+-+-+-+-+-+-+-+               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               |  Pad Length   | Next Header   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Authentication Data (variable)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPSec - overview

- IPSec basic features
- IPSec transport and tunnel modes.
- AH – authentication and data integrity.
- ESP – confidentiality.
- IPSec policy and Security Associations.
- Combining Security Associations
- Key management in IPSec: ISAKMP and IKE
- NAT traversal
- Use case: VPN
    - “SSL VPNs”

IPSec Basic Features

- IPSec provides security at network (Internet) layer.
  - So all IP datagrams covered.
  - No re-engineering of applications.
  - Transparent to users.
- Mandatory for next-generation IPv6, optional for current-generation (IPv4).
- Defined in IETF RFCs 4301–4309 .

IPsec as a standard: RFC4301-4309

RFC 4301 Security Architecture for the Internet Protocol
RFC 4302 IP Authentication Header
RFC 4303 IP Encapsulating Security Payload (ESP)
RFC 4304 Extended Sequence Number (ESN) Addendum to IPsec Domain of Interpretation (DOI) for Internet Security Association and Key Management Protocol (ISAKMP)
RFC 4305 Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH)
RFC 4306 Internet Key Exchange (IKEv2) Protocol
RFC 4307 Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2)
RFC 4308 Cryptographic Suites for IPsec
RFC 4309 Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)

Documents are at: https://www.ietf.org/rfc/rfc430x.txt, x=1,2,…,9

IPsec as a standard (cont'd)

- RFC 2407 - The Internet IP Security Domain of Interpretation for ISAKMP
  - This RFC describes the use of ISAKMP — Internet Security Association and Key Management Protocol — in the context of IPSec. It's a framework for key exchange at the start of a conversation, and its use obviates the poor practice of using manual keys.
- RFC 2408 — Internet Security Association and Key Management Protocol (ISAKMP)
  - Hand in hand with RFC 2407, this RFC dives into much more detail on the ISAKMP protocol used to support key exchange (though it doesn't define the key exchange protocols themselves).
- RFC 2409 — The Internet Key Exchange (IKE)
  - Though ISAKMP provides a framework for key-exchange, it doesn't define the protocols themselves: this RFC does that. IKE includes initial authentication, as well as Oakley key exchange.
- RFC 2410 — The NULL Encryption Algorithm and Its Use With IPsec

IPsec as a standard (cont’d)

- All these are very technical
- Good reference “IPSec” by N. Doraswamy and D. Harkins (Prentice Hall, 1999).
- strongSWAN project: http://www.strongswan.org/

IPSec Basic Features

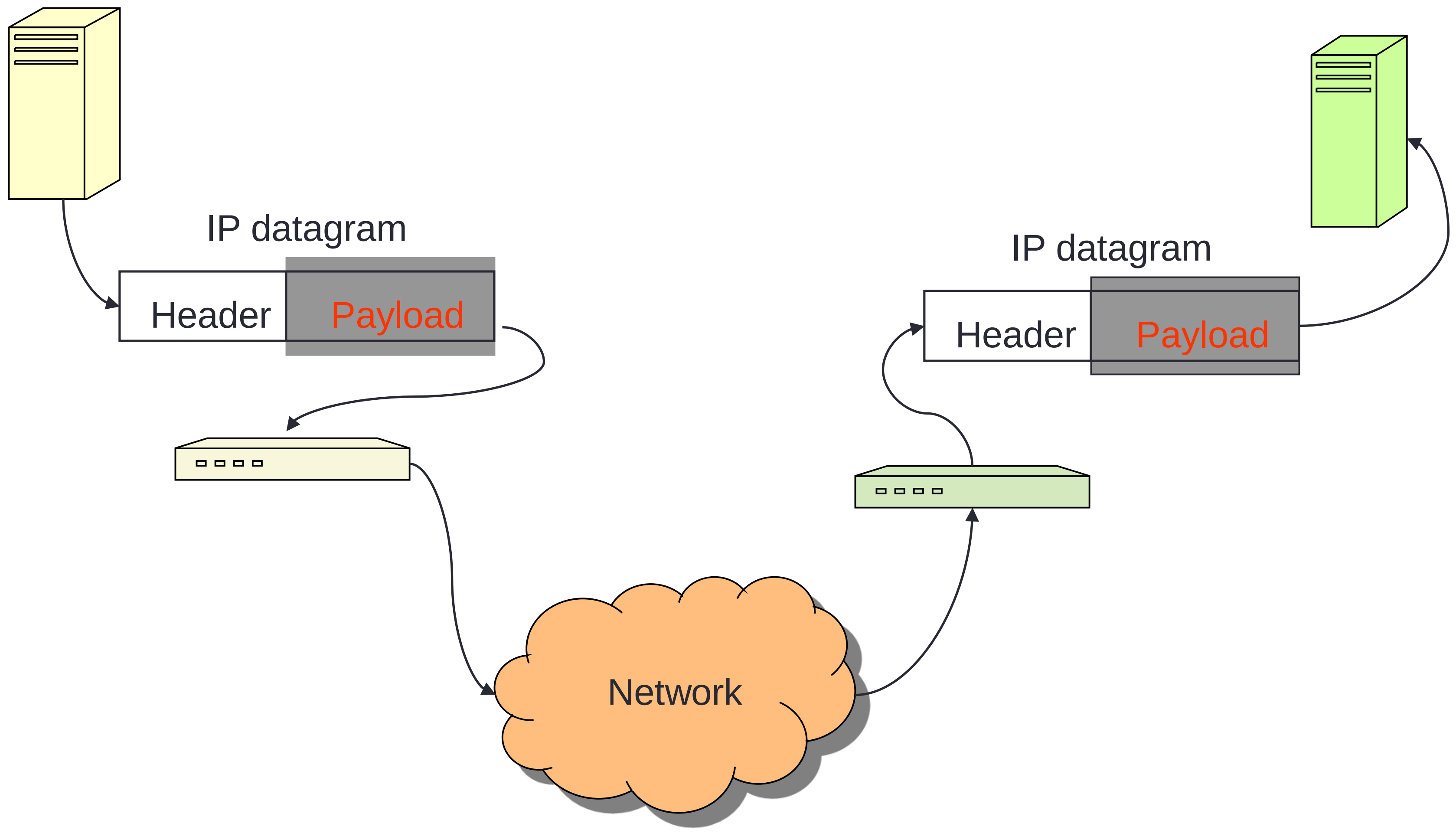
- IPSec provides two basic modes of use:
  - “transport” mode: for IPSec-aware hosts as endpoints.
  - “tunnel” mode: for IPSec-unaware hosts, established by intermediate gateways or host OS.

- IPSec provides authentication and/or confidentiality services for data with two protocols
  - AH and ESP protocols.

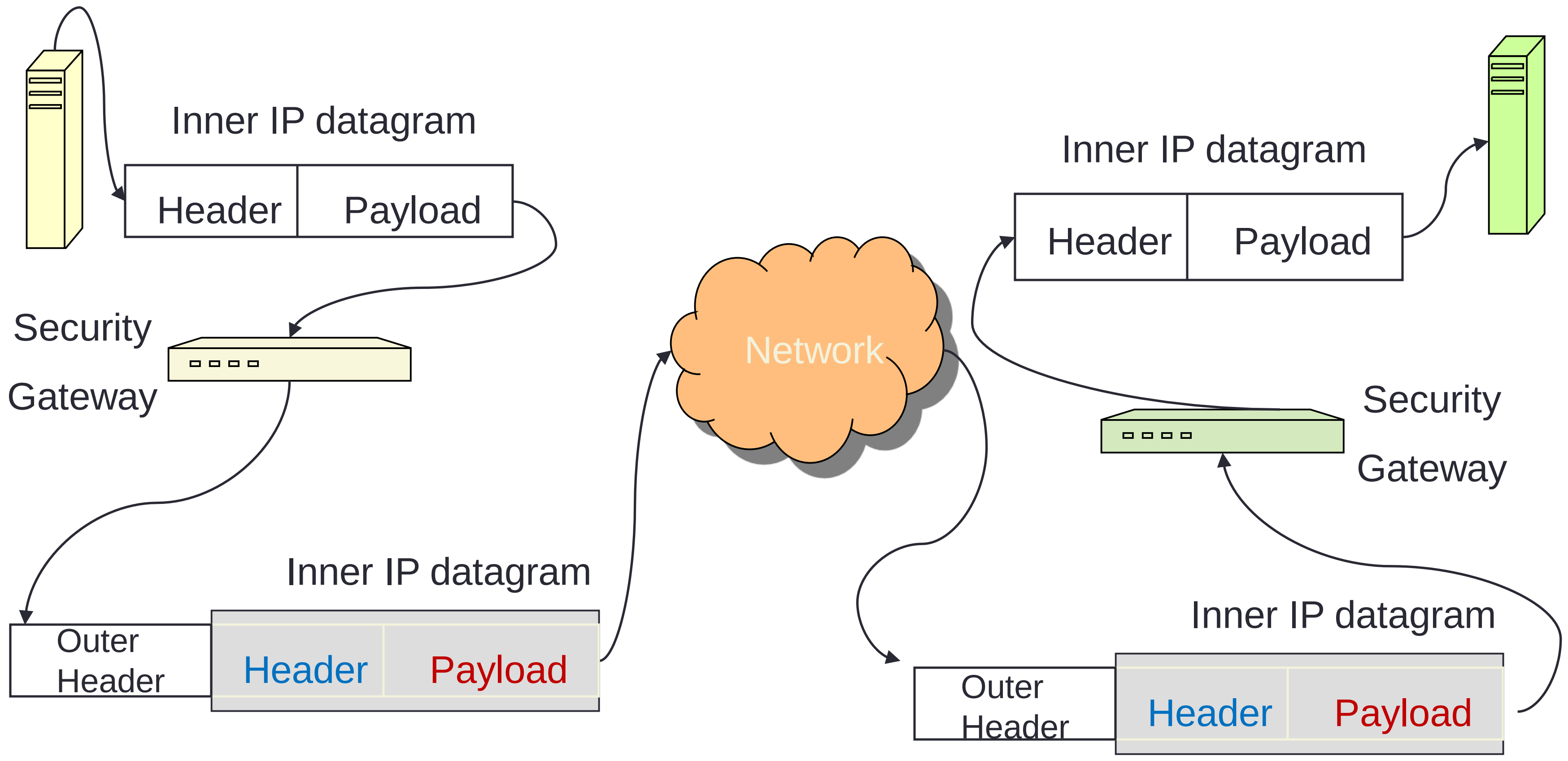
Key establishment for IPSec

- IPSec itself lacks key establishment but the “IPSec” standards provide a flexible set of key establishment methods:
  - IKE (derived from ISAKMP and Oakley) which now is IKEv2
  - all this is very complex !!!
  - Instead of IKE vendors use proprietary solutions
    - Results in inoperable systems (even if IPsec is used)

IPSec Transport Mode

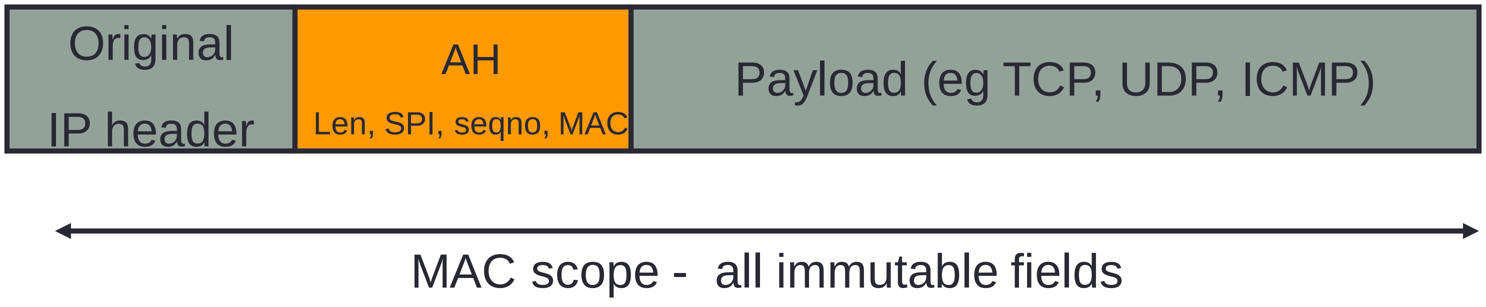


IPSec Tunnel Mode

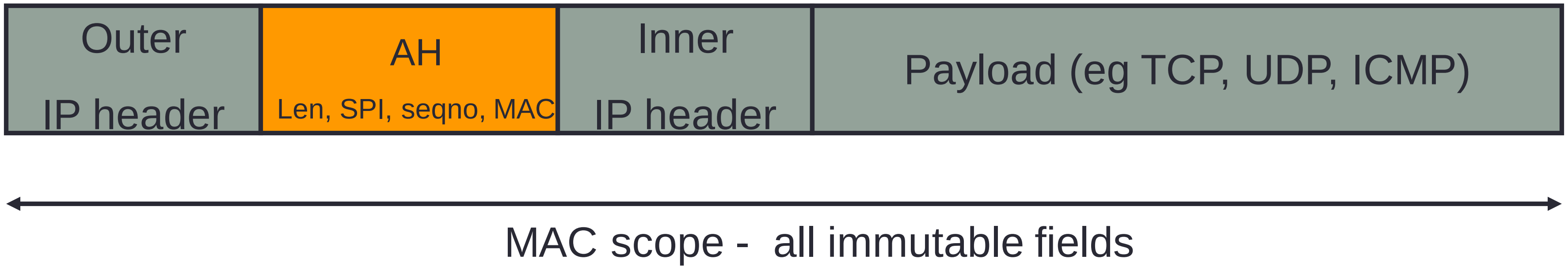


AH Protocol – Transport and Tunnel

AH in transport mode:

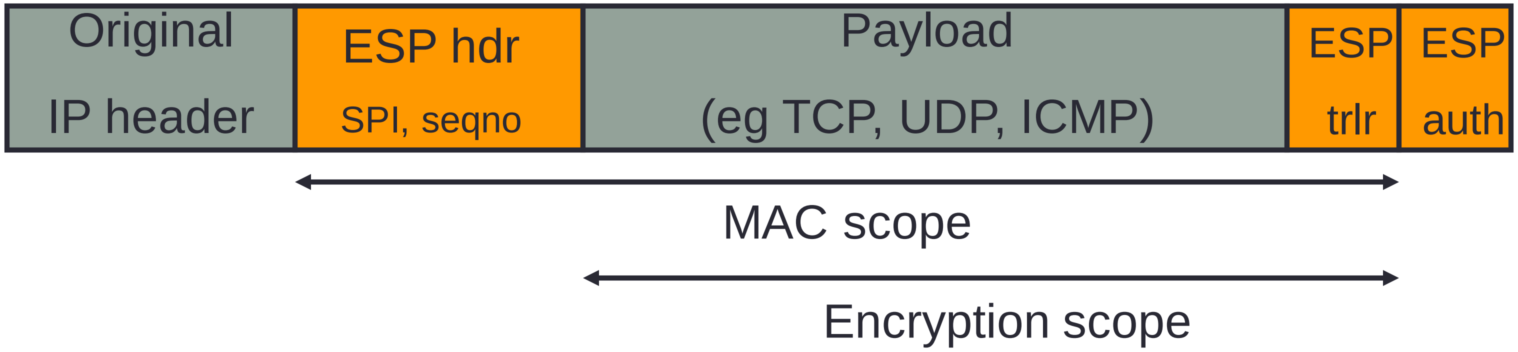


AH in tunnel mode:

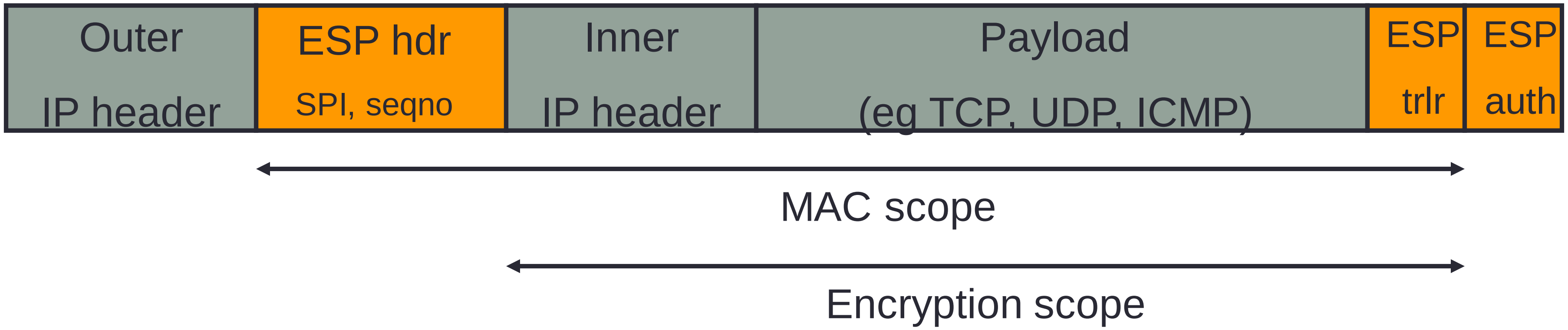


ESP Protocol – Transport and Tunnel

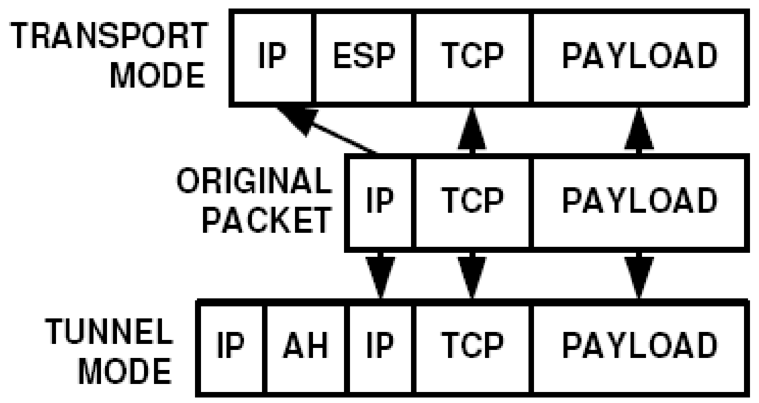
ESP in transport mode:

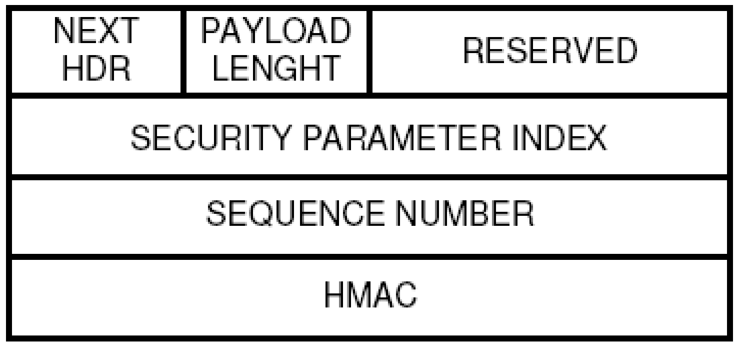


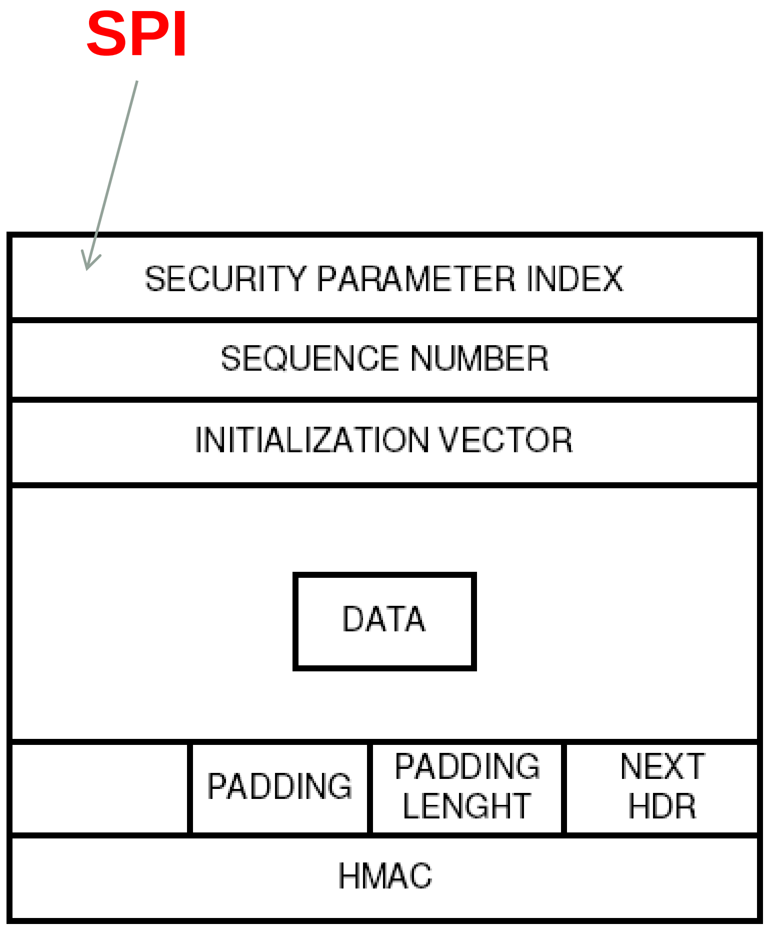
ESP in tunnel mode:



IPSec processing and packet formats







AH and ESP Algorithms

- IPSec supports the use of a number of algorithms for ESP and AH.
- ESP:
  - Null (mandatory)
  - DES, three-key triple DES CBC (mandatory),
  - AES-CBC with 128-bit key (recommended)
  - AES-CTR (recommended)
  - HMAC-SHA1-96 (mandatory)
- AH:
  - HMAC-SHA1-96 (mandatory)
  - AES-XCBC-MAC-96 (recommended)

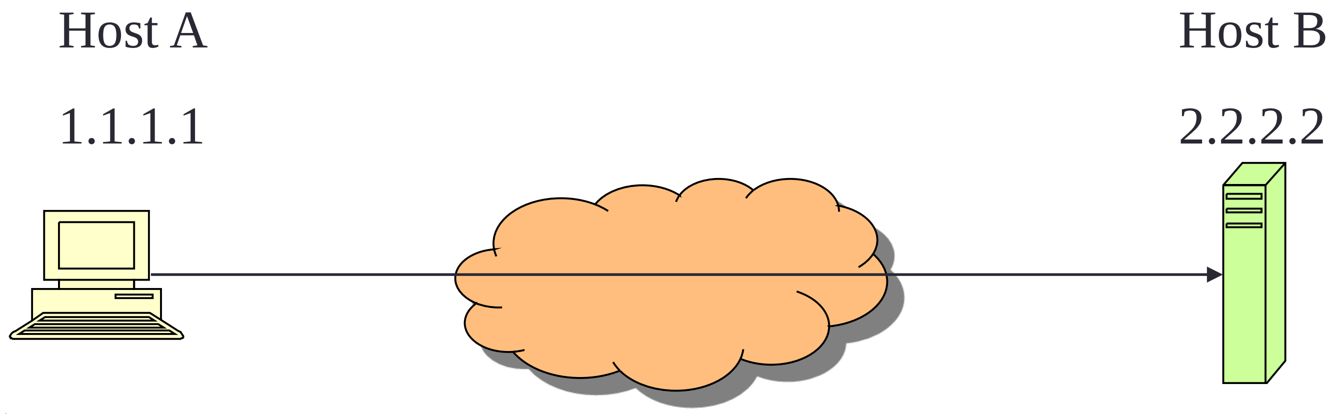
Sequence Numbers in IPSec

- Both ESP and AH use sequence numbers to provide an anti-replay service.
- Sequence numbers are 32 bits long.
  - Initialised to zero.
  - Increment on datagram-by-datagram basis.
  - Overflow results in auditable event and re-keying.
- Protected by MACs in AH and ESP
  - But no protection afforded to sequence numbers when ESP (confidentiality only) is used.
- Recipient uses “sliding window” to track datagram arrivals.
  - Recommended window length is 64.
  - Datagrams can be dropped if delayed too long (by network latency or deliberately).

IPSec Security Associations (SAs)

- An SA is a one-way (simplex) relationship between sender and receiver.
  - Specifies cryptographic processing to be applied to *this* datagram from *this* sender to *this* receiver.
- SAs are held in a SA database (SADB)
  - list of active SAs
- Each SA is identified by unique SPI (32 bit value carried in AH and ESP headers).
  - Allows recipient to determine how to process received datagrams.
- Each SA contains:
  - Sequence number counter and anti-replay window.
  - AH/ESP info: algorithms, IVs, keys, key lifetimes.
  - SA lifetime.
  - Protocol mode: tunnel or transport.
  - …

SPDs and SAs in Action



A’s SPD:

From	To	Protocol	Port	Policy
1.1.1.1	2.2.2.2	TCP	80	Transport ESP with 3DES

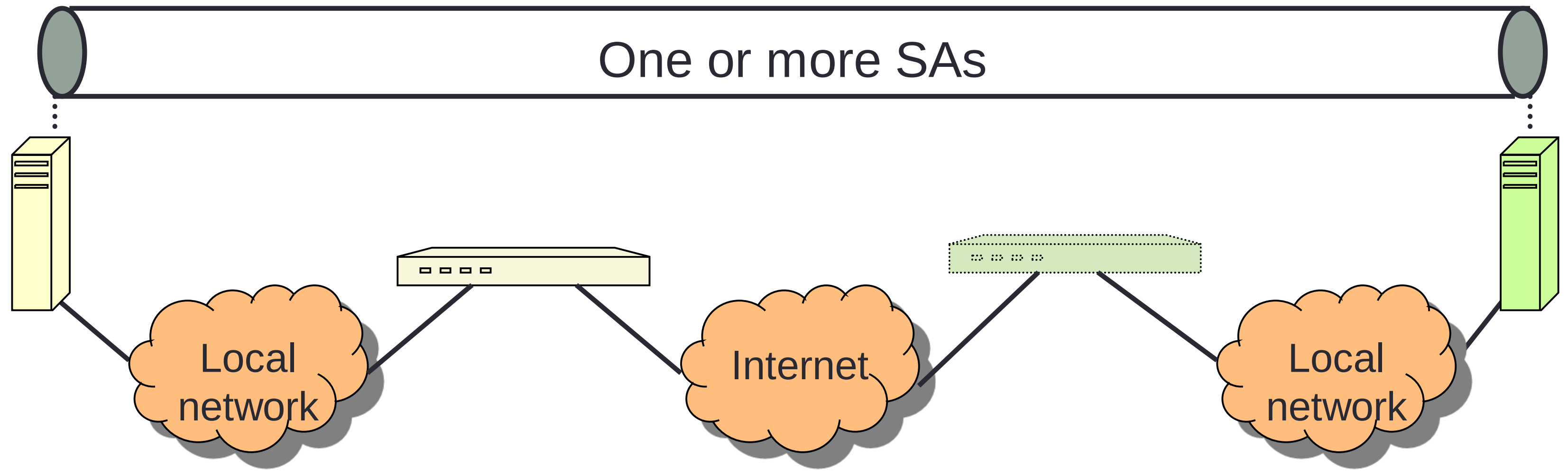
A’s Outbound SADB:

From	To	Protocol	SPI	SA record
1.1.1.1	2.2.2.2	ESP	10	3DES key

Required SA Combinations

End-to-end application of IPSec between IPSec-aware hosts:

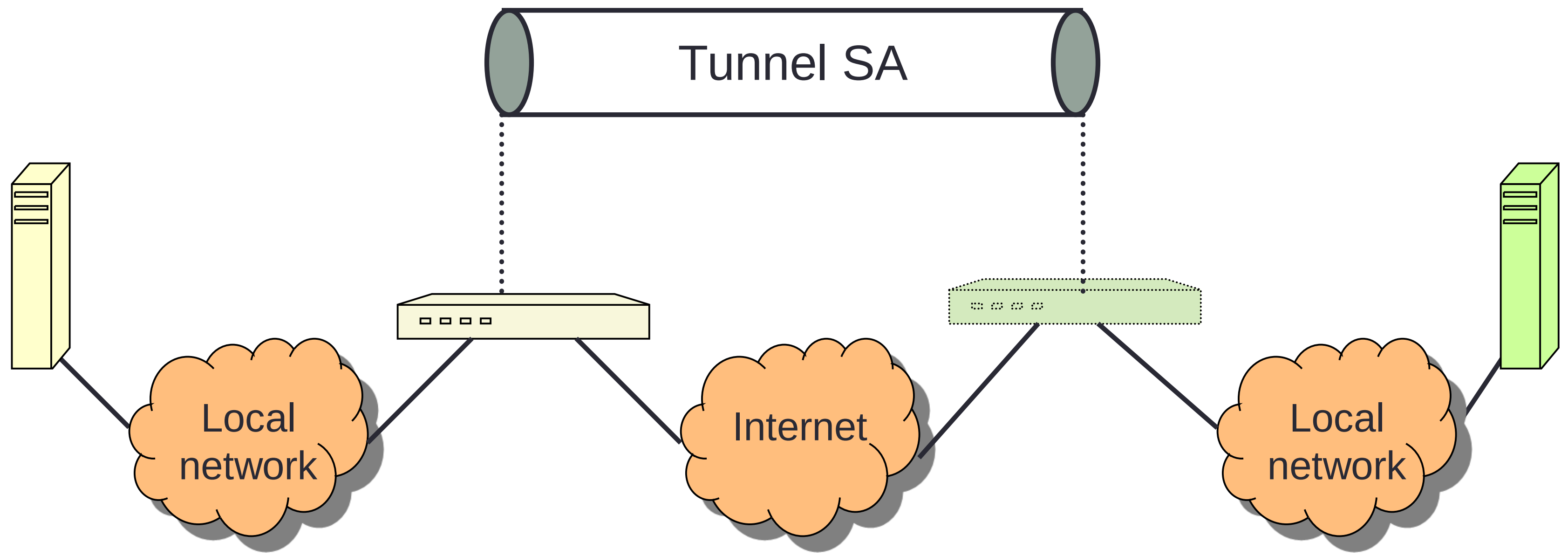
- One or more SAs, one of the following combinations:
  - AH in transport
  - ESP in transport
  - AH followed by ESP, both transport
  - Any of the above, tunnelled inside AH or ESP.



Required SA Combinations

Gateway-to-gateway only:

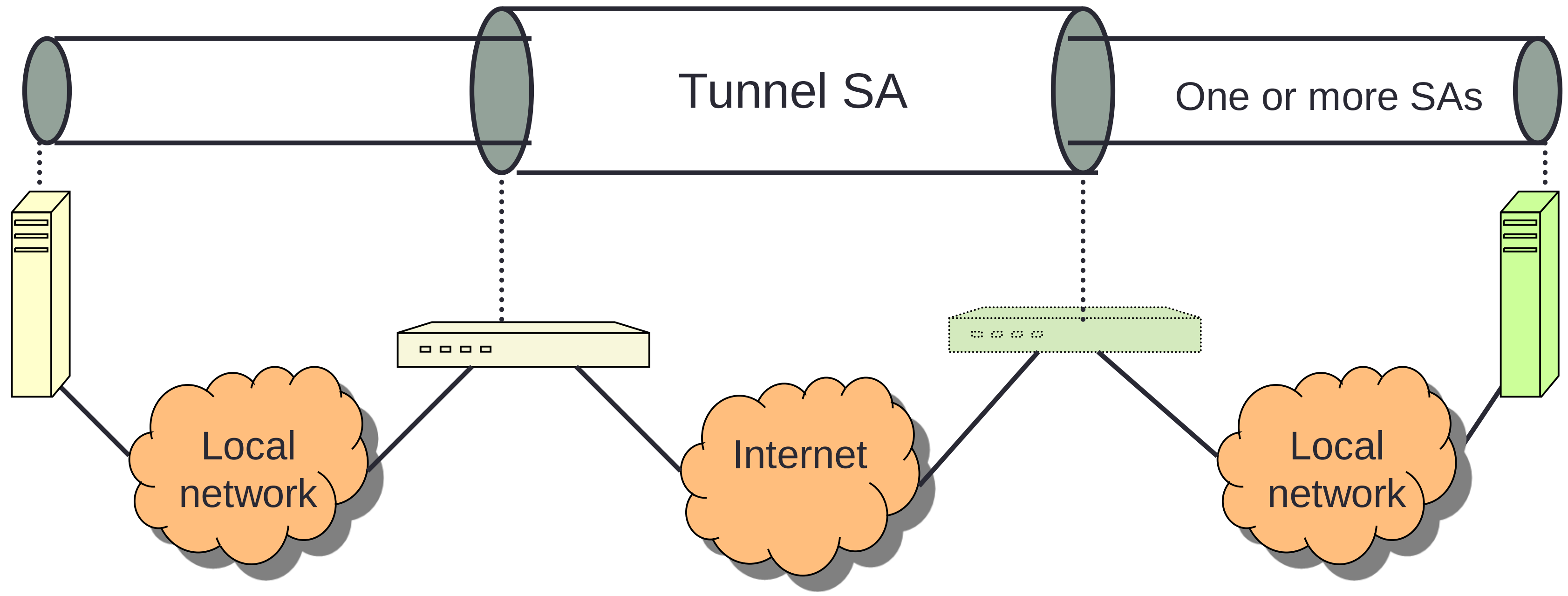
- No IPSec at hosts.
- Simple Virtual Private Network (VPN).
- Single tunnel SA supporting any of AH, ESP (conf only) or ESP (conf+auth).



Required SA Combinations

A combination of the two previous ones:

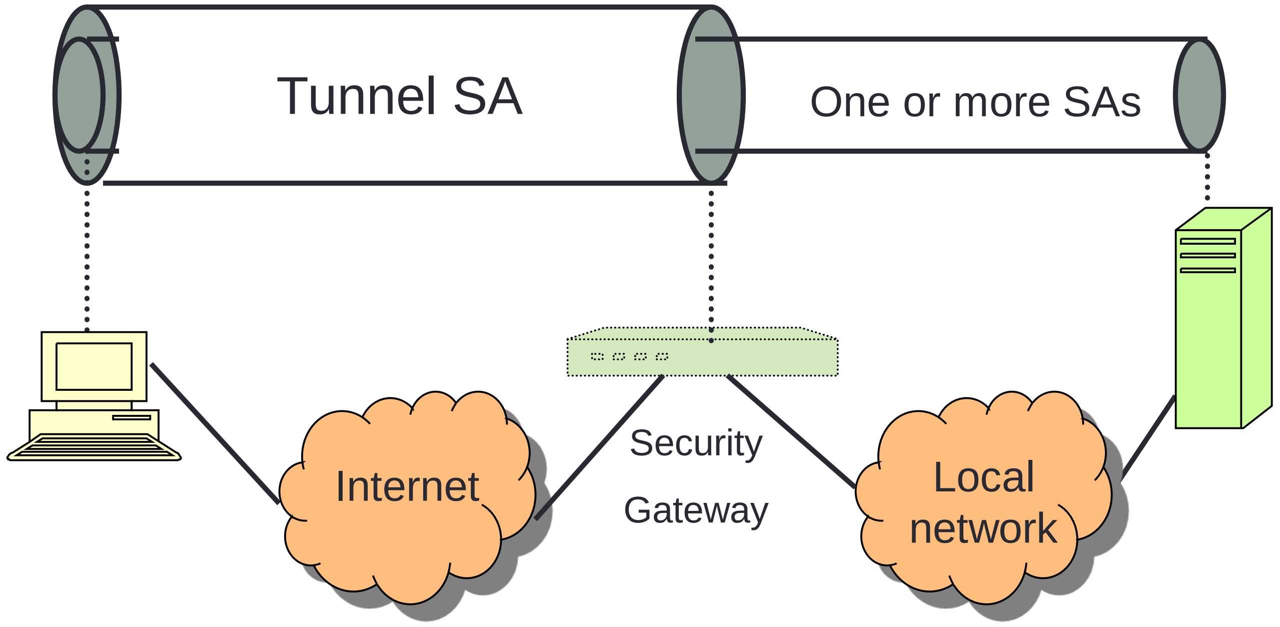
- Gateway-to-gateway tunnel as in 2 carrying host-to-host traffic as in 1.
- Gives additional, flexible security on local networks (between gateways and hosts)
- E.g., ESP in tunnel mode carrying AH in transport mode.



Required SA Combinations

Remote host support:

- Single gateway (typically firewall).
- Remote host uses Internet to reach firewall, then gains access to server behind firewall.
- Traffic protected in inner tunnel to server as in case 1 above.
- Outer tunnel protects inner traffic over Internet.



IPSec Key Management

- IPSec is a heavy consumer of symmetric keys:
  - One key for each SA.
  - Different SAs for:
    - {ESP,AH} x {tunnel,transport} x {sender, receiver}.
- Where do these SAs and keys come from? -> Two options
  - Manual keying.
    - Fine for small number of nodes but hopeless for reasonably sized networks of IPSec-aware hosts; requires manual re-keying.
  - IKE: Internet Key Exchange, IKEv1 and IKEv2 (RFC 4306).
    - RFC documentation hard to follow.
    - IKE is a specific adaptation of more general protocols ("Oakley" and "ISAKMP").
    - Protocols have many options and parameters.

IKE Security Goals

- Entity authentication of participating parties.
- Establishment of a fresh, shared secret.
  - Shared secret used to derive further keys.
  - For confidentiality and authentication of IKE management channel.
  - For SAs for general use.
- Resistance to Denial-of-Service attacks.
  - Using cookie mechanism.
- Secure negotiation of all algorithms.
  - Authentication method, key exchange method, group, algorithms for encryption and MAC, hash algorithms.
- Options for Perfect Forward Secrecy, Deniable Authentication and Identity Protection.

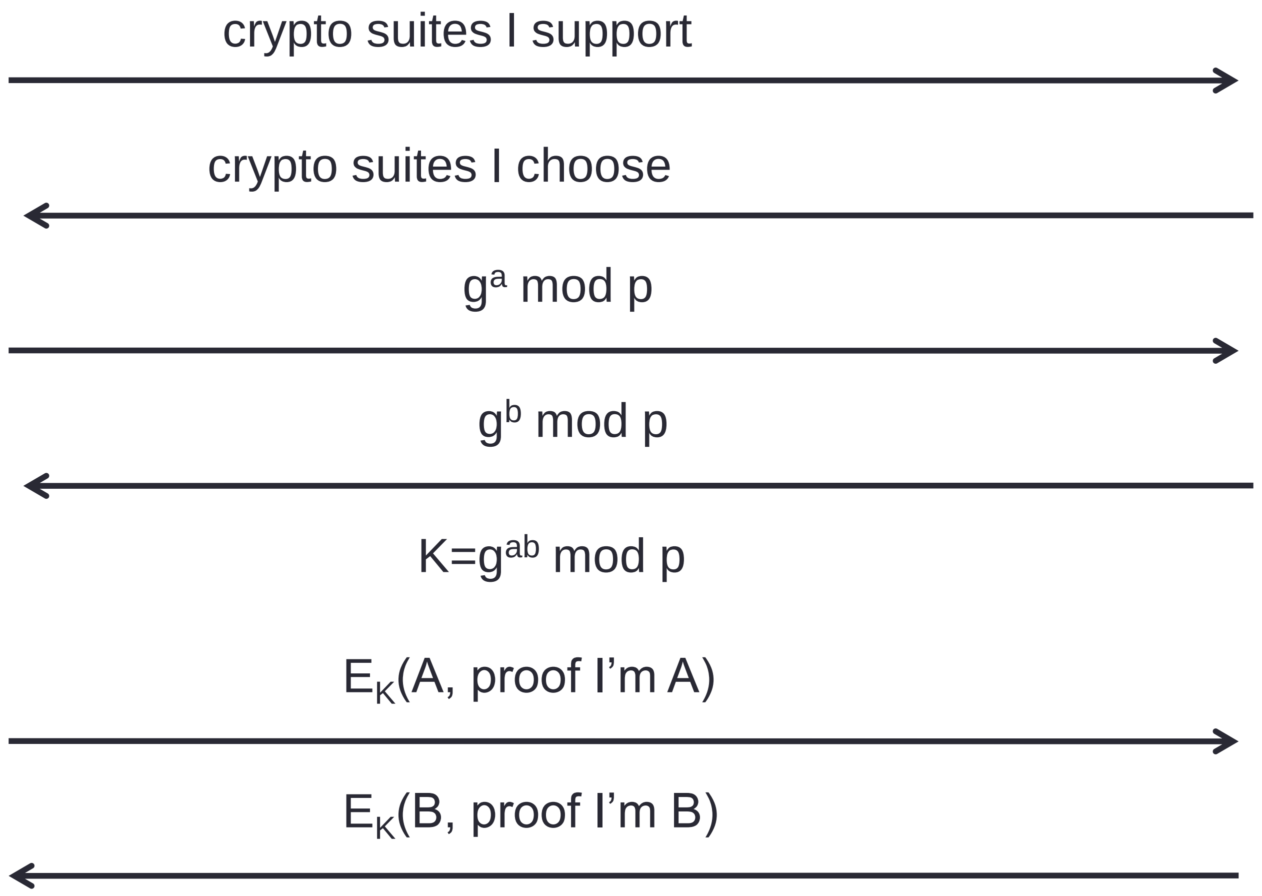
IKE Phases

IKE operates in two phases.

- **Phase 1** provides mutual authentication and establishes session key

  Set up an SA and secure channel to carry further SA negotiation, as well as error and management traffic.
  - Bi-directional.
  - Heavy-duty entity authentication and key exchange.
  - Establishes ISAKMP channel (IPSec key management protocol) – a secure channel for use in Phase 2.
- **Phase 2** allows for multiple security associations for same Phase 1 pair. SAs for general use are negotiated.
  - Fast negotiation takes place over Phase 1 secure channel.
  - Many Phase 2 runs allowed for each run of Phase 1.
  - Multiple SAs can be negotiated per run.

Phase 1 IKE Main Mode



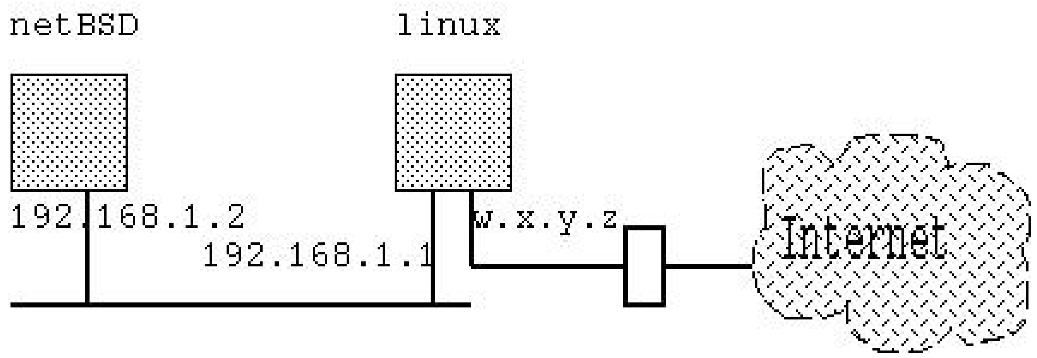
Notes on IPSec

- IKE is carried over UDP; hence unreliable and blocked by some firewalls.
- IPSec and firewalls have problems working together.
  - Authentication of source IP addresses in AH is the issue.
  - Some firewalls change these addresses on out-bound datagrams.
- Managing IPSec policy and deployments is complex.
  - Getting it wrong can mean losing connectivity, e.g. by making exchanges of routing updates unreadable.
  - Getting it wrong can mean loss of security.
  - Many, many IPSec options, poor documentation.
- Microsoft has put IPSec into WinXP, replacing PPTP.
- IPSec now part of standard Linux distribution.
- IPSec is mandatory part of IPv6

IPSEC AND NAT TRAVERSAL

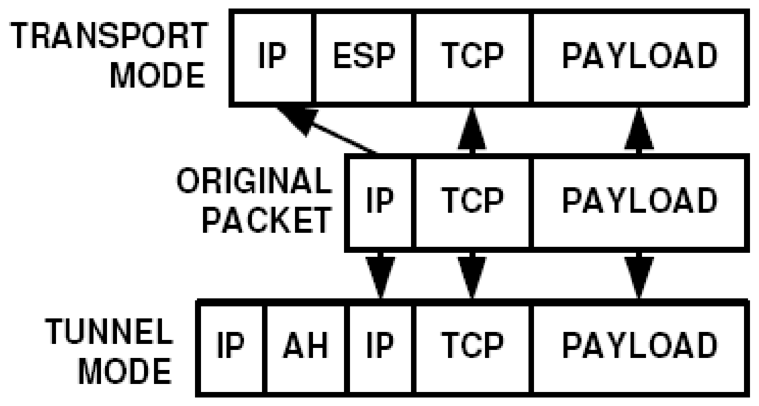
Parts of this material has been compiled from various open sources

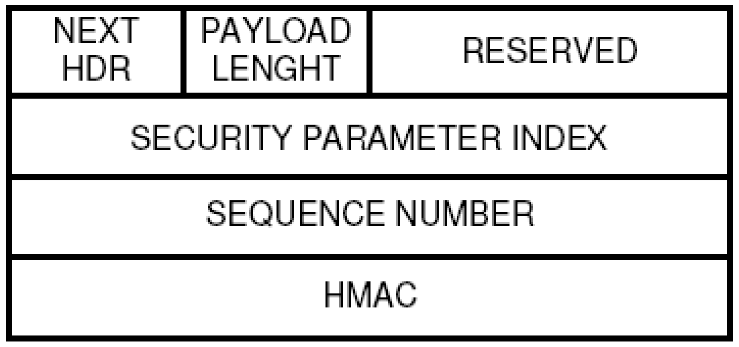
Network Address Translation

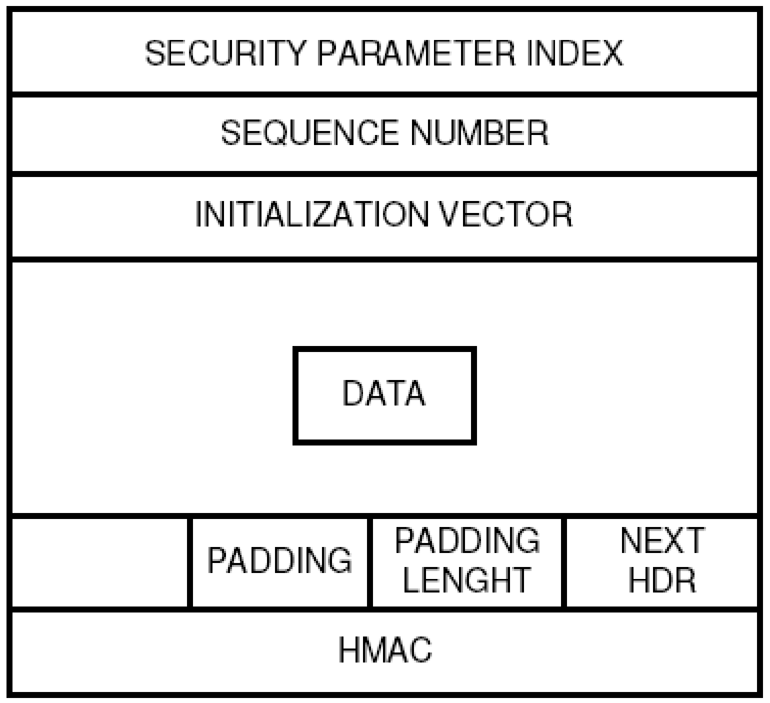


- Basic NAT
  - 192.168.1.2 → w.x.y.z
- NAPT
  - 192.168.1.2, Source Port A →w.x.y.z, Source Port B

IPSec processing and packet formats







IPSEC- NAT Incompatibility

- AH header incorporates the IP source and destination address for integrity check.
- TCP/UDP checksums.
- IKE-NAT Incompatibilities.

NAT-T: UDP encapsulation of IPsec ESP packets

- ESP: Only payload is encrypted

⇒ NAT-T adds a UDP header that encapsulates the ESP header.

Functionality: (during initial IPSec negotiation)

1. If peers have NAT-T capability
2. NAT router in the middle of the path between the peers

⇒ Otherwise normal IPsec operations

IKE and NAT (well this is strictly not IPsec but it is used together)

IKE:

- Internet Key Exchange for IPsec
  - 1st phase: SA and key exchange protocol (ISAKMP) establishes the a secure authenticated channel for further negotiation traffic, and defines the SA used during negotiations.
  - 2nd phase: SA is negotiated used by IPsec.
- Normal IKE traffic is performed over UDP to port 500.

- Non-ESP-marker field that allows a recipient to distinguish between UDP encapsulated ESP PDU and an IKE message.

- IKE includes new payloads
  - Vendor ID: hash value (indicates the capability for NAT-T)
  - NAT-OA (Original Address)

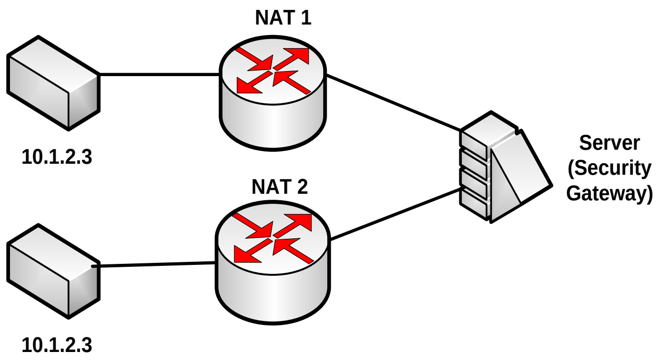
NAT-T SOLUTIONS

1) A receiving peer gets all required information for verification process of upper-layer checksum (IKE payload: NAT-OA payload).
2) A receiving peer has the original IP address where it can verify the contents of the identification IKE payload during quick mode negotiation.
3) IPsec peers can accept IKE messages from different source port than 500 -> IKE UDP port 4500 is used.
4) NAT router uses the UDP ports for multiplexing of the IPsec data streams.
5) NAT-T introduces keep alive messages.

NAT-T PROBLEMS

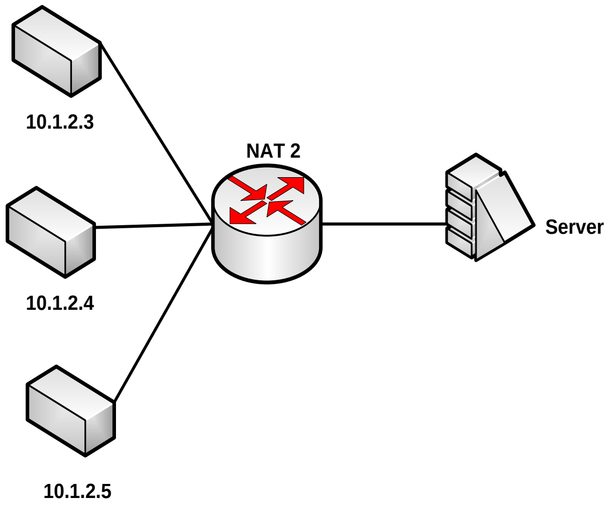
- **Tunnel mode conflict**

  Remote peers may negotiate entries that overlap when tunnel mode is used.



- **Transport mode conflict**

  May occur when two peers behind NAT routers are in communication with same server. Server may get confused which SA is belonging to which client.



CONCLUSIONS

- AH incompatible, ESP can be used.
- NAT-T solution uses ESP
  - UDP/TCP
  - IPv6

- NAT-T working solution with some problems.
  - PATH: Client->NAT->Internet->Server
    - Only supported model

- OS may have NAT-T disabled as default.