

# Security (CDK chapter 7)

## I Know I'm Paranoid, But am I paranoid enough 😊

Security is a process, not an event!

Policy, Mechanisms, Assurance, Response and Education

Who are you trying to protect what from and why?

- SWOT analysis
- Security requirements engineering
- Penetration testing
  
- Confidentiality (privacy)
- Authentication (who created or sent the data)
- Integrity (has not been altered)
- Non-repudiation (the order is final)
- Access control (prevent misuse of resources)
- Availability (permanence, non-erasure)
- Denial of Service Attacks
- Virus that deletes files
  
- **Interruption:** This is an attack on availability
- **Interception:** This is an attack on confidentiality
- **Modification:** This is an attack on integrity
- **Fabrication:** This is an attack on authenticity

# Cryptography 101

- We don't need to be able to design or analyse algorithms!
- Cryptosystems generally fail because of incorrect USE, not design.
- We simply use well understood & tested algorithms generally as part of standard protocols.
- Kerckhoffs Principle (1883):
  - The security of a cryptosystem should depend on the key alone, not the algorithm.

## ***Kerckhoff in detail:***

1. The system must be practically, if not mathematically, indecipherable;
2. It must not be required to be secret, and it must be able to fall into the hands of the enemy without inconvenience;
3. Its key must be communicable and retainable without the help of written notes, and changeable or modifiable at the will of the correspondents;
4. It must be applicable to telegraphic correspondence;
5. It must be portable, and its usage and function must not require the concurrence of several people;
6. Finally, it is necessary, given the circumstances that command its application, that the system be easy to use, requiring neither mental strain nor the knowledge of a long series of rules to observe.

**Reference:** Auguste Kerckhoffs, *La cryptographie militaire*, Journal des sciences militaires, vol. IX, pp. 5-83, Jan. 1883, pp. 161-191, Feb. 1883.

# The big problem!

- Communicating a shared secret!
- Traditional symmetric block ciphers (e.g. DES or IDEA) are effective and relatively computationally cheap.

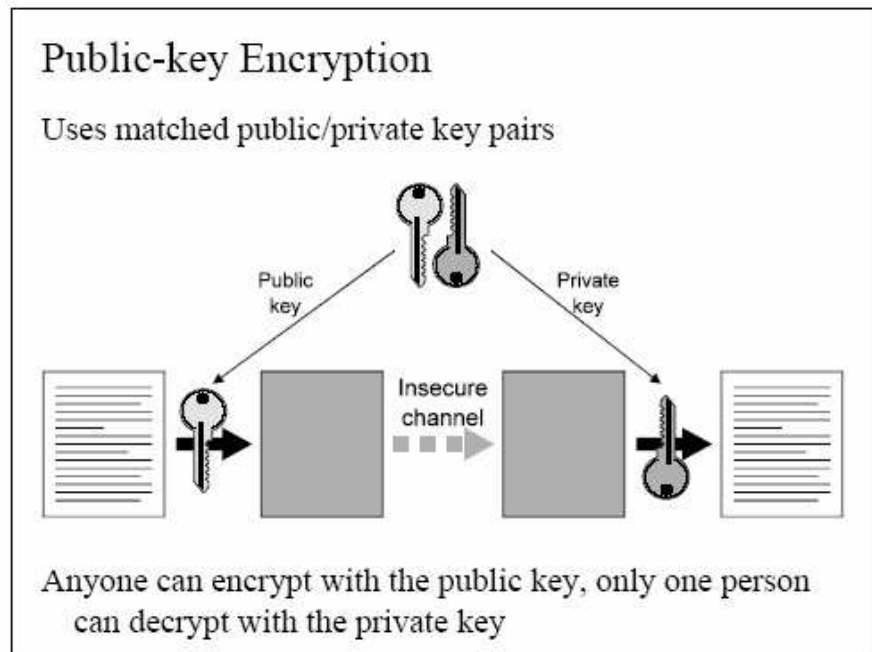
From Stallings, "Network Security"

Key Size (bits)	Number of Alternative Keys	Time required at 106 Decryption/ $\mu$ s
32	$2^{32} = 4.3 \times 10^9$	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	10 hours
128	$2^{128} = 3.4 \times 10^{38}$	$5.4 \times 10^{18}$ years
168	$2^{168} = 3.7 \times 10^{50}$	$5.9 \times 10^{30}$ years

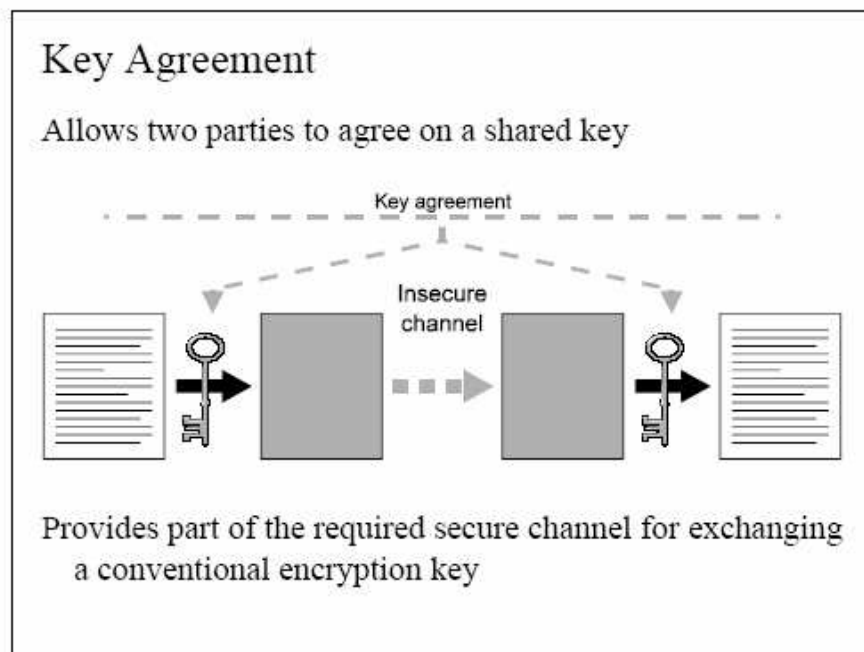
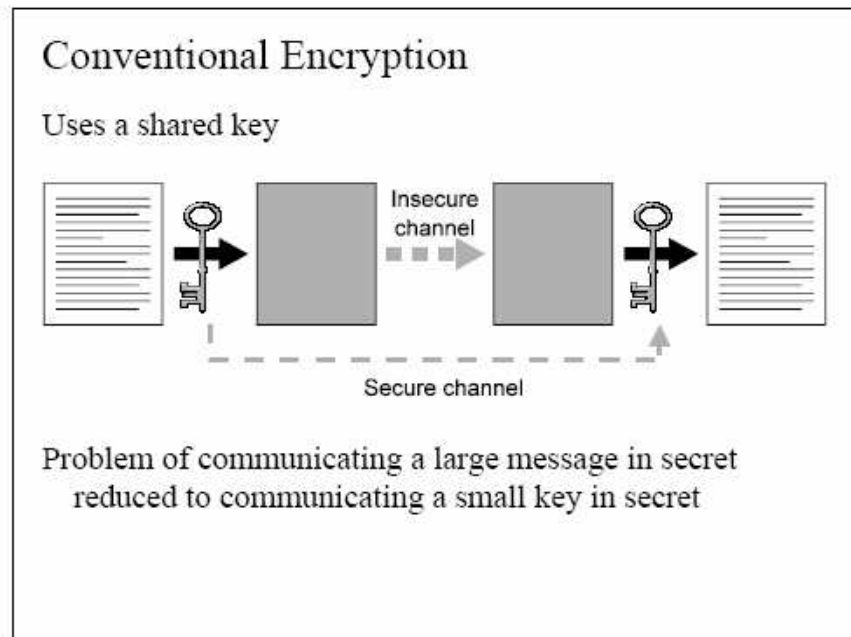
- But:  
We need to communicate the key!
- Most modern systems (SSL, PGP/GPG) use an asymmetric cipher (e.g. RSA) to communicate or negotiate a symmetric key. This is often used for a single session, in which case it is a session key.

# The tools

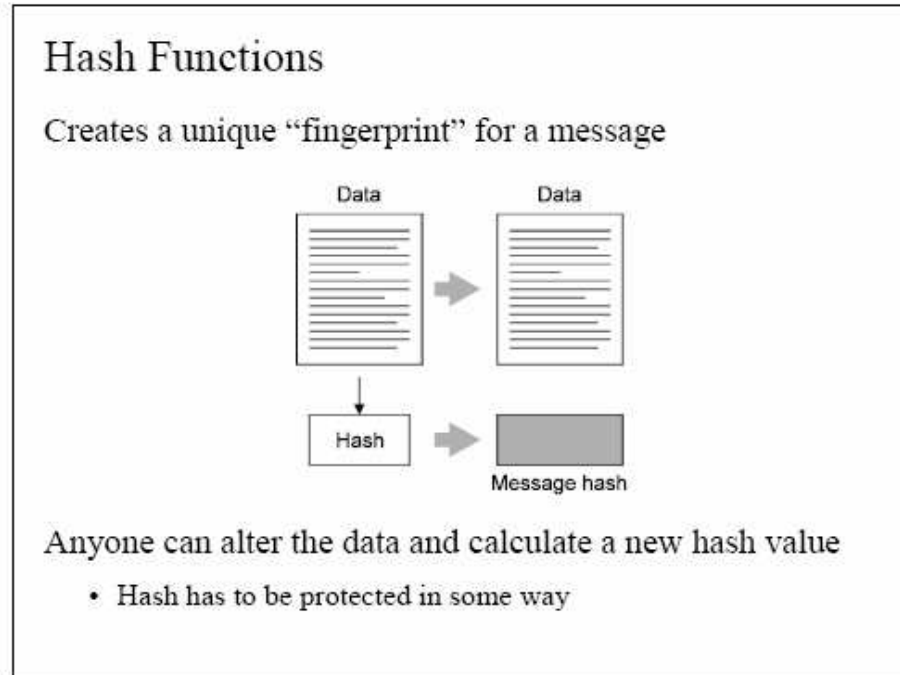
- Asymmetric (a.k.a. Public key) algorithms
  - Computationally expensive
  - Can in theory be broken eventually



- Symmetric algorithms  
Key needs to be communicated securely



- Message Digests
  - A hash function (MD5 is a well known example)
  - Generates a difficult to forge “fingerprint” for a file
  - 1 in 18,446,744,073,709,551,616



## The goals

- Privacy
- Authentication

In reality, any system will always have more requirements

# Public Key encryption

Public key encryption relies on the difficulty of factoring very large numbers, and testing numbers for primeness.

As a result, given sufficient time, any public key algorithm can be broken, but the time required depends on key length:

But all problems that interest public key cryptographers are either P or NP.

- P = can be solved in polynomial time, e.g. product of 2 numbers of  $n$  bits can be solved in at most  $n^2$  bit operations
- All problems that are P are also NP, but (it is believed) not the reverse! A problem is NP if a solution can be checked in polynomial time. Factoring a number is NP.

Diffie & Hellman published the first description of Public Key in 1976.

It had been independently discovered in the late 1960's by Ellis, Cox & Williamson at GCHQ. Some reports were declassified in 1997.

All PK algorithms are based on (presumed) **Trapdoor One Way Functions**. One way functions are functions that are easy to compute one way and hard the other. Trapdoor one way functions have a trapdoor that (if you know it) makes the reverse direction easy too.

Most modern cryptosystems use factoring as the “hard” problem, although Elliptic-curve algorithms hold some promise, since currently no non-exponential time solution is known.

Size of problem is directly proportional to key length.

RSA Challenges have now demonstrated the 512 bit is weak.

Increasing the key length by 1 bit doubles the effort required to brute force the key. Doubling key length therefore increases effort required by something of the order of  $2^{512}$ .

Doubling the key length makes decrypt operations around four times slower and encrypt operations around eight times slower.

*From RSA Security FAQ ([www.rsasecurity.com](http://www.rsasecurity.com))*

	Block Cipher	RSA	Elliptic Curve	DSA
Export Grade	56	512	112	512 / 112
Traditional recommendations	80 112	1024 2048	160 224	1024 / 160 2048 / 224
Lenstra/Verheul 2000	70	952	132	952 / 125
2010	78	1369	146 / 160	1369 / 138

Table 2. Minimal key lengths in bits for different grades.

Some key points:

1. Public key compromises are often very serious!
2. Digital signatures may be required to persist for a very long time!
3. What is secure today, may not be tomorrow, will not be in 100 years.
4. Bit length is not a good indicator of strength **on its own**. All things being equal, it may well be.
5. Paranoia is important!
6. OTPs usually are not 😊
7. Expertise requires research on security engineering. Let others do the crypto.

Lets have a look at RSA:

1. Select 2 large (>1024 bit) prime numbers P & Q
  - a.  $p = 17, q = 11$
2. Compute product (N)
  - a. 187
3. Select E where:
  - a.  $E > 1,$
  - b.  $E < PQ$
  - c. E and  $(P-1)(Q-1)$  have no common prime factors
  - d. We'll use  $E = 7$
4. Ciphertext ( C ) = Message (M)<sup>E</sup> (mod N)

We want to transmit ASCII X (88 decimal)  
N & E are the public key

$$C = 88^7 \pmod{187}$$

$$C = 11$$

When this is received, it cannot be easily decrypted without knowledge of P & Q. Using these values we can compute the private key (D).

$$E * D = 1 \pmod{(p-1) * (q-1)}$$

$$7 * D = 1 \pmod{16 * 10}$$

$$D = 23$$

To decrypt we compute:

$$M = C^D \pmod{187}$$

$$M = 11^{23} \pmod{187}$$

$$M = 88$$

# Public Key Properties

If we know  $(E,N)$  the public key, we can encrypt so that decryption requires  $(D,N)$ .

If we know  $(D,N)$  the private key, we can encrypt so that decryption requires  $(E,N)$ , we can use this with a message digest as a digital signature. This allows us to authenticate.

Authentication in this case means matching a public and private key, to match to identity is harder and requires some sort of CA infrastructure.

## Cracking Encryption

In 1998 under the direction of John Gilmore of the EFF, a team spent \$220,000 and built a machine that can go through the entire 56-bit DES key space in an average of 4.5 days. On July 17, 1998, they announced they had cracked a 56-bit key in 56 hours. The computer, called Deep Crack, uses 27 boards each containing 64 chips, and is capable of testing 90 billion keys a second.

More recently, in early 1999, Distributed.Net used the DES Cracker and a worldwide network of nearly 100,000 PCs to win the RSA DES Challenge III in a record breaking 22 hours and 15 minutes. The DES Cracker and PCs combined were testing 245 billion keys per second when the correct key was found. In addition, it has been shown that for a cost of one million dollars a dedicated hardware device can be built that can search all possible DES keys in about 3.5 hours. This just serves to illustrate that any organisation with moderate resources can break through DES with very little effort these days.

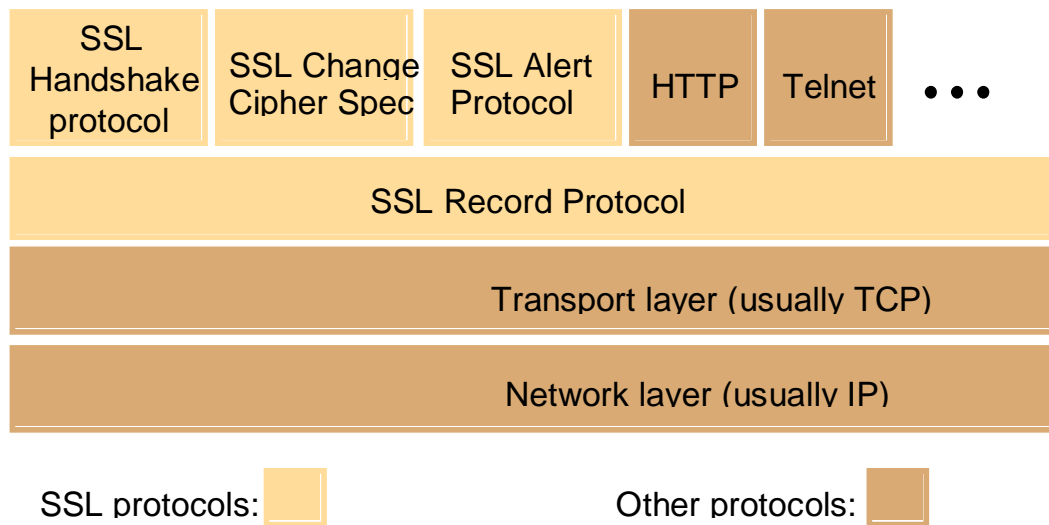
# Where do we use encryption?

- Encrypted Email (S/MIME, GPG, PGP, etc.)
  - key security?
- Digital signatures (GPG, PGP)
  - We also use a digital signature to protect our tripwire fingerprint.
- SSH – host/client verification, encrypted channel
- SSL – encrypted channel, certificate based identification.
  - Standard use for secure HTTP
  - Can be used as a wrapper for any socket based service.
  - Supported by the openssl library.
- Kerberos – encrypted network traffic

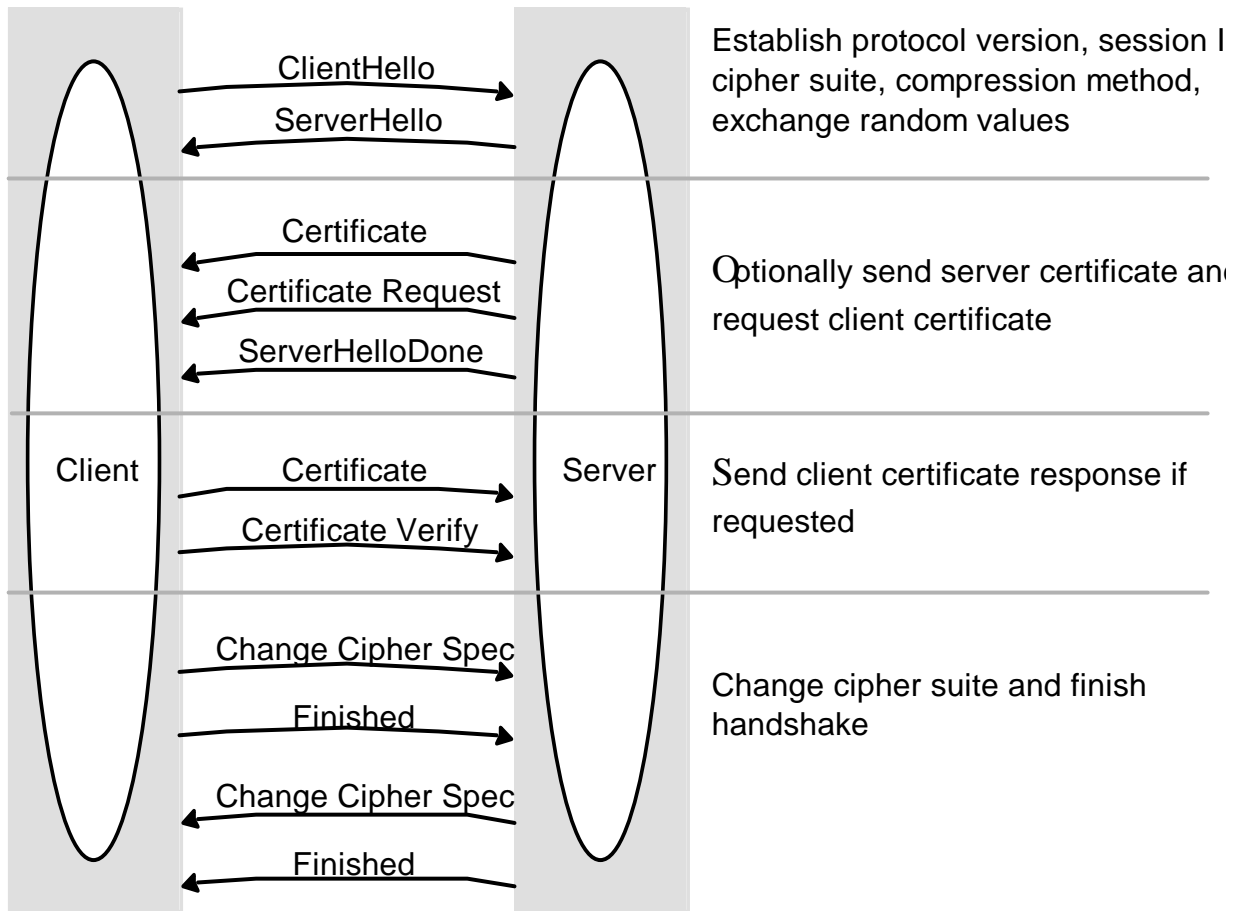
# The Secure Sockets Layer (SSL)

- SSL is also known as TLS
- Supported on most architectures
- Supported by an open source library (openssl)
- Two part tutorial on the module web page

From CDK: 7.17, SSL protocol stack



SSL Handshake (from CDK fig. 7.18)



## SSL Handshake configuration options (CDK fig 7.19)

<i>Component</i>	<i>Description</i>	<i>Example</i>
Key exchange method	the method to be used for exchange of a session key	RSA with public-key certificates
Cipher for data transfer	the block or stream cipher to be used for data	IDEA
Message digest function	for creating message authentication codes	SHA

X509 certificates;

- **X509 ITU standard adopted by IETF and widely used on the Internet**
- 
- **IETF version of X509 is RFC 3280**
- 
- **X509 is a way to describe certificates (meta- data or schema written in ASN.1 )**

<b>Field</b>	<b>Meaning</b>
Version	Which version of X.509
Serial number	This number plus the CA's name uniquely identifies the certificate
Signature algorithm	The algorithm used to sign the certificate
Issuer	X.500 name of the CA
Validity period	The starting and ending times of the validity period
Subject name	The entity whose key is being certified
Public key	The subject's public key and the ID of the algorithm using it
Issuer ID	An optional ID uniquely identifying the certificate's issuer
Subject ID	An optional ID uniquely identifying the certificate's subject
Extensions	Many extensions have been defined
Signature	The certificate's signature (signed by the CA's private key)

# An alternative to SSL – IPsec

## Result of an argument over Internet security

- Security mechanisms should be end to end hence in the applications
- But users don't understand security & all applications will have to be changed
- OK next best thing put it into the Transport protocol or just above it.
- Well lets put it in the network because it will help the security naïve

## The complete Ipsec is a complete a la carte framework

- Multiple services, algorithms & granularities
- RFCs 2401, 2402, 2406 (RFC 2410 the null algorithm)

## Major services

- Secrecy, integrity, protection from replay
- Symmetric key cryptography for performance
- Algorithm independent so new ones can be introduced in the future
- Connection oriented – Security Association

## 2 Major Parts

- 2 new headers added to IP packets
- Internet Security Association & Key Management Protocol (ISAKMP)

# IPSec modes

## Transport Mode

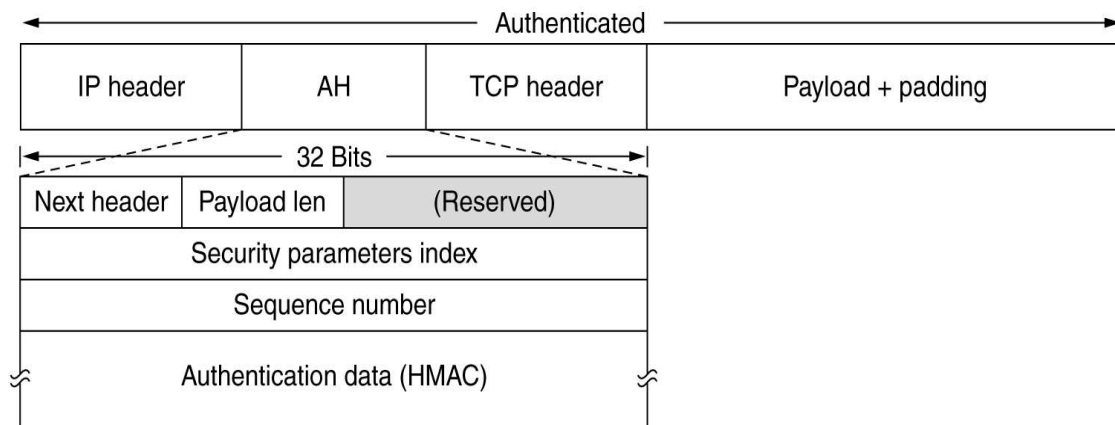
- The Ipsec header is inserted just after the IP header

## Tunnel Mode

- The entire IP packet header and all is encapsulated in the body of the new packet with a new IP header
- This is ideal when the two end points are firewalls. So between secure networks the firewalls can encapsulate/decapsulate packets as they pass through the firewall. This means company machines don't have to worry about Ipsec.
- A number of TCP connections can be bundled encrypted & encapsulated in one packet. This can hide the number of communications.

# IPSec authentication header in Transport level (IPv4)

NOTE: No Encryption!

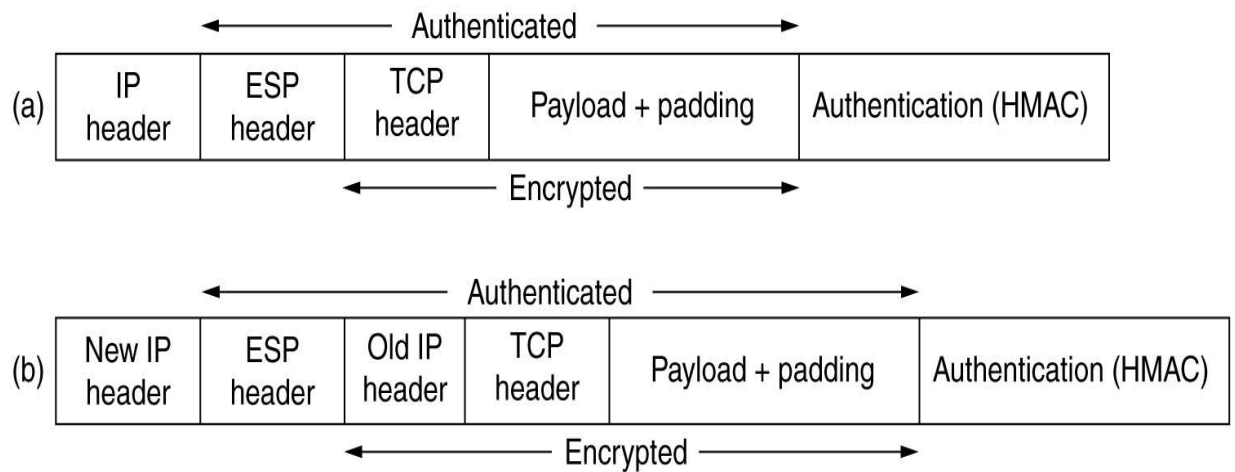


# Encapsulating Security Payload (ESP)

a) transport mode

b) tunnel mode

Note: ESP will probably make AH obsolete



## *Security Bibliography*

Obviously, chapter 7 from CDK! However if your interest has raised in this area, the following are very useful references.

Anderson, Ross; “*Security Engineering*”, Wiley, 2001

Stallings, William; “*Network Security Essentials*”, 2000

Singh, Simon; “*The Code Book*”, 1999

RSA Inc.; “*RSA Security FAQ 4.1*” ([www.rsasecurity.com](http://www.rsasecurity.com))

Ferguson, Niels & Schneier, Bruce; “*Practical Cryptography*”, 2003

Mitnick, K.D. & Simon, W.L.; “*The Art of Deception*”, 2002