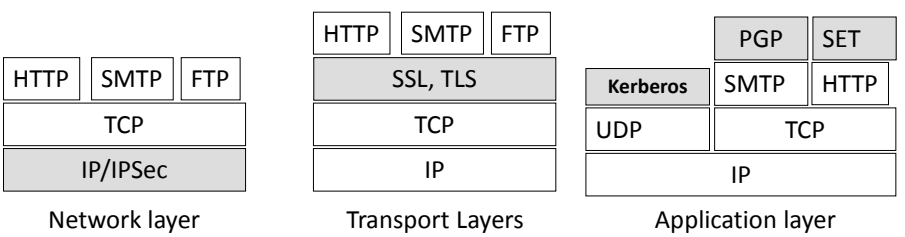
Secure Socket Layer (SSL)

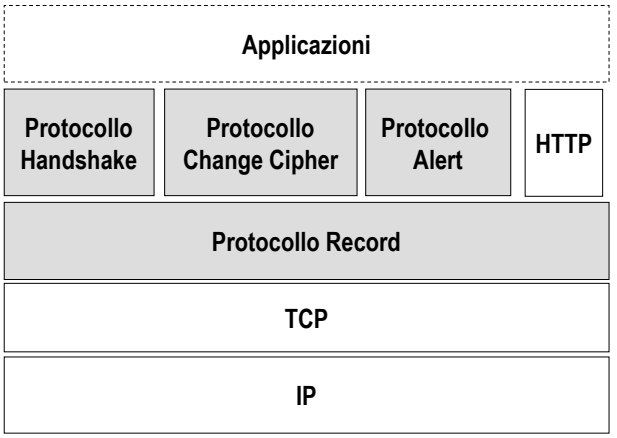
SSL

THE PROTOCOL

Security in the TCP/IP stack



La suite di protocolli SSL



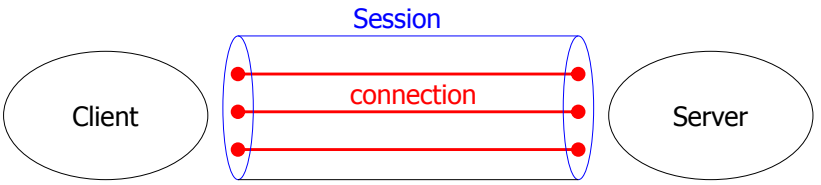
References

- **Secure Socket Layer (SSL)**
  - Netscape
  - http://wp.netscape.com/eng/ssl3/
- **Transport Layer Security (TLS)**
  - Based on SSL v3.0
  - RFC 2246
  - ftp://ftp.rfc-editor.org/in-notes/rfc2246.txt
  - Same design as SSL but different algorithms

History of the protocol

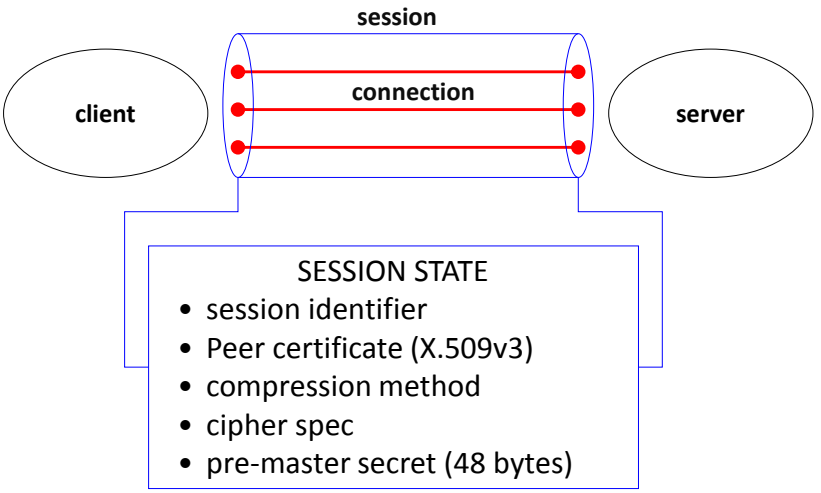
- **SSL**
  - Developed by Netscape in mid 1990s
  - SSLv1 broken at birth (never publicly released)
  - SSLv2 flawed, now IETF-deprecated (RFC 6176)
  - SSLv3 still widely supported (since 1996)
- **TLS**
  - IETF-standardized version of SSL.
  - TLS 1.0 in RFC 2246 (1999), based on SSLv3 but NOT interoperable
  - TLS 1.1 in RFC 4346 (2006).
  - TLS 1.2 in RFC 5246 (2008).

Session vs connection

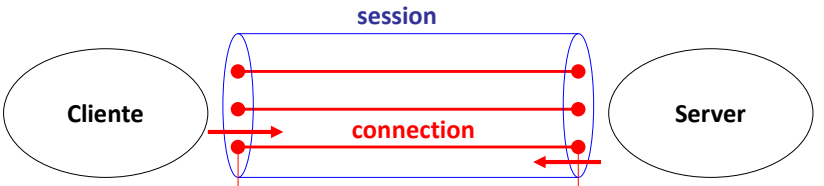


- A **session** is a logical association between a Client and a Server
  - Created by the **Handshake protocol**
  - Define a set of **crypto pars** that can be shared by multiple connections
  - Avoid **expensive** negotiation of crypto pars for each connection

Session vs connection



Session vs connection



CONNECTION STATE

- Server random number (nonce)
- Client random number (nonce)
- Server write MAC secret
- Client write MAC secret
- Server write key
- Client write key
- Initialization vectors
- Sequence numbers

The Record Protocol

Payload

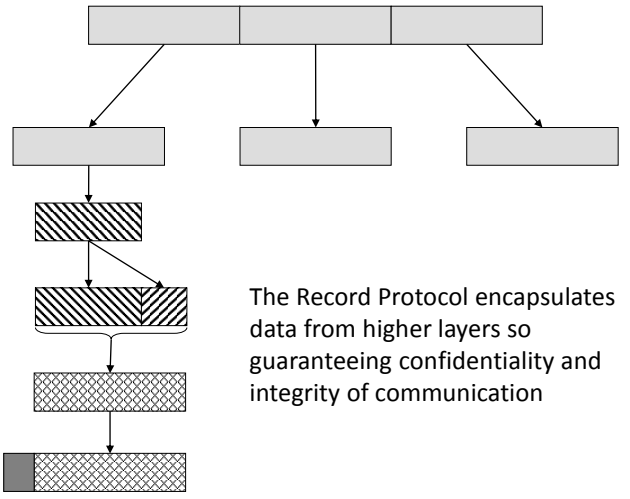
Fragmentation
(max 2^{14} bytes)

Compression
max 2^{14} + 1024 bytes

MAC

Encryption

Heading
(max 2^{14} + 2048)



The Record Protocol encapsulates data from higher layers so guaranteeing confidentiality and integrity of communication

The Record Protocol

- **Fragmentation** fragments application data in blocks whose size is $\leq 2^{14}$-bytes
- **Compression** must be lossless and must not increase the block size more than 1024 bytes (default = null)
- MAC uses **the [Server|Client] write MAC key**, sequence number, compressed block, padding
- Encryption uses the **[Server|Client] write key**
  - Block and steam ciphers
  - Does not increases the content size more than 1024 byte
- Total length of a fragment must be $\leq 2^{14} + 2048$ bytes

SSL **09/05/2018** **11**

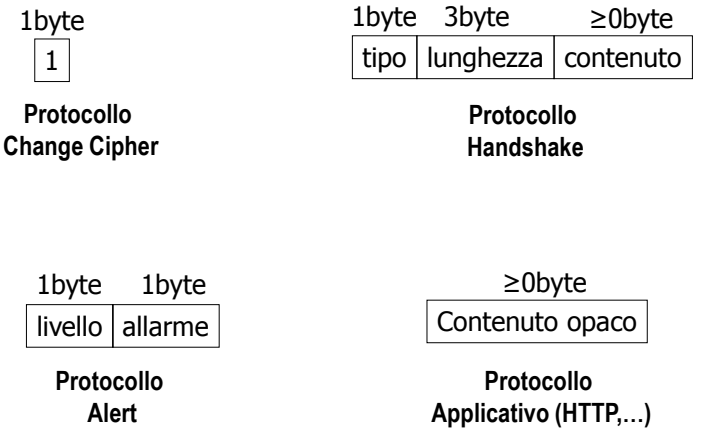
The Record Protocol

Header

- **Payload type**
  - change cipher, alert, handshake, application_data
  - Application data is opaque to SSL
- **Major Version**
  - SSLv3 => 3
- **Minor Version**
  - SSLv3 => 0
- **Compressed length**
  - Fragment size $\leq 2^{14} + 2048$

SSL **09/05/2018** **12**

Payload types

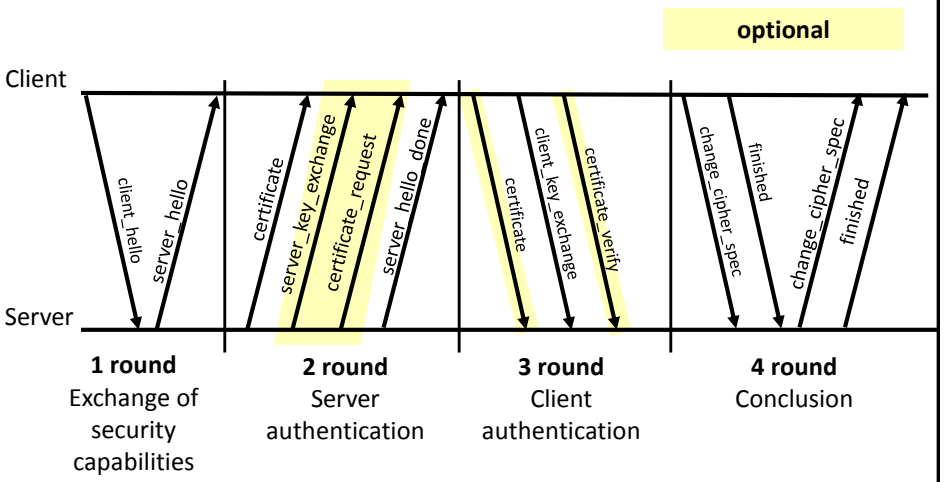


The other protocols in the SSL suite

- The **change cipher spec protocol** consists in one single message (cleartext) to make the negotiated crypto suite operational
- The **alert protocol** notifies alarms to peers

FATAL ALARMS	OTHER ALARMS
unexpected_message	no_certificate
bad_record mac	bad_certificate
decompression_failure	unsupported_certificate
handshake_failure	certificate_revoked
illegal_parameter	certificate_expired
	certificate_unknown
	close_notify

Handshake protocol: an overview



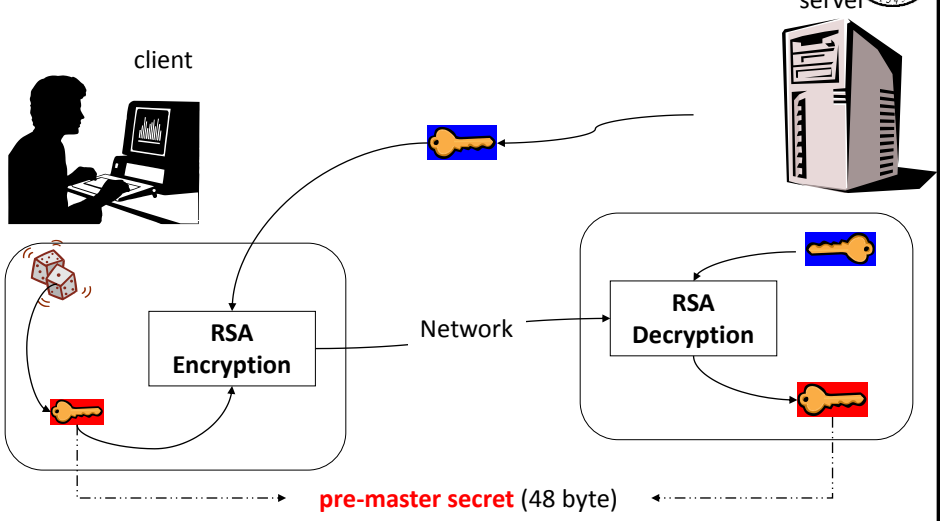
Set of messages

TIPO	CONTENUTO
hello_request	No pars
client_hello	version, nonce, session Idcipher suite, compression method
server_hello	version, nonce, session id, cipher suite, compression method
certificate	Certificate X.509v3
server_key_exchange	Pars, signature
certificate_request	Type, authority
server_hello_done	No pars
certificate_verify	signature
client_key_exchange	Pars, signature
finished	hash

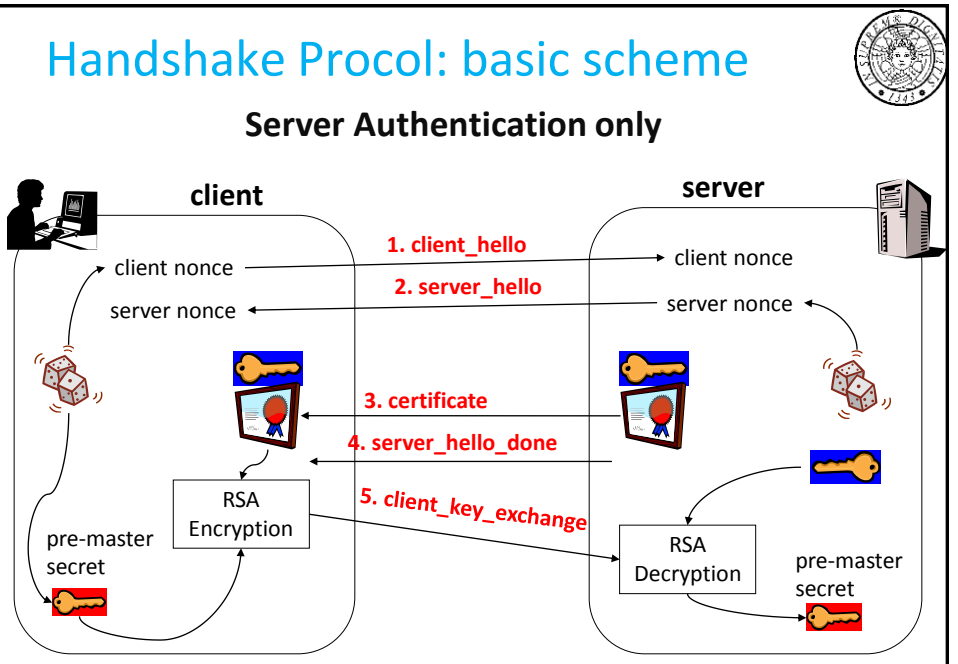
The Handshake Protocol

- Establish a secure session
  - Client and server authenticate each other
  - Client and server negotiate the cipher suite
    - Key establishment scheme;
    - Encryption scheme (used in the RP)
    - MAC (used in the RP)
  - Client and server establish a shared secret
    - E.g., pre-master secret
- Before any application data
- The most complex part of SSL

Handshake Protocol: basic scheme



Handshake Procol: basic scheme



Hello message

- By means of **Hello** msgs, Client and Server tell each other what they are able to do
  - client_hello and server_hello
  - SSL version
  - Random: timestamp [32 bit] + random bytes[28]
  - Session id
  - Cipher suite
  - Compression method

Cipher suite

- Cipher suite is a list of algorithm *tuples*
- A *tuple* specifies
  - Key establishment
  - cipher, cypher type, IV size, isExportable
  - MAC, hash size
  - key material
- Some tuples are standard
  - E.g., SSL_RSA_WITH_3DES_EDE_CBC_SHA

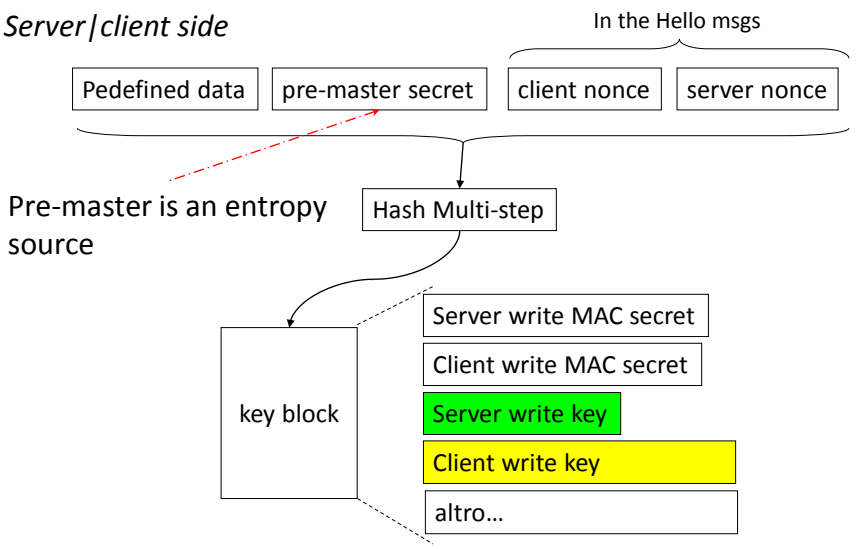
Cipher suite

- Supported key establishment schemes
  - RSA (certified)
  - Fixed Diffie-Hellman (certified; fixed pub pars)
  - Ephemeral Diffie-Hellman (signed, dynamic pub pars)
  - Anonymous Diffie-Hellman (non authenticated)
- Supported ciphers
  - RC4, RC2, DES, 3DES, IDEA, …
- Supported MAC
  - MD5, SHA-1

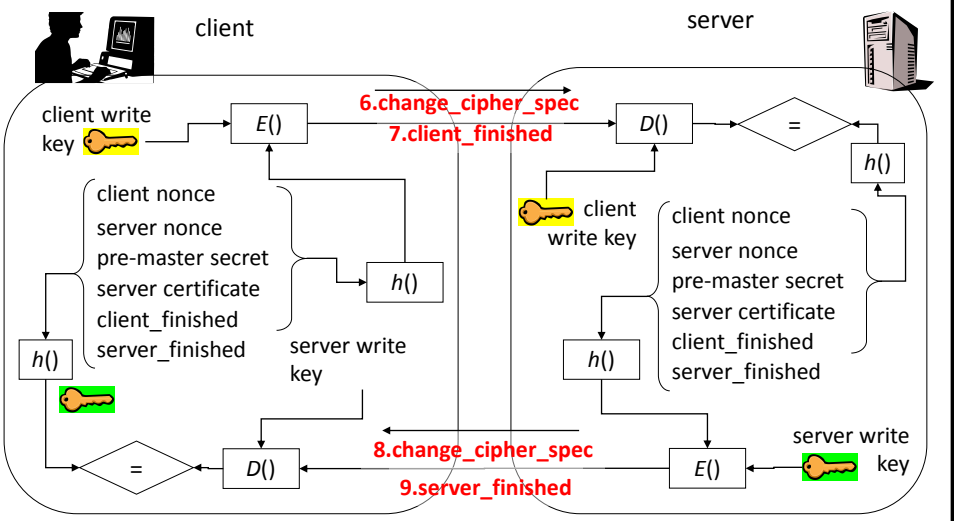
Client_key_exchange message

- The message format depends on the chosen key establishment
  - **RSA** – pre-master secret
  - **Anonymous or ephemeral DH** – $(p, g, Y)_{clnt}$
  - **Fixed DH** – void payload, public pars will be sent in a **certificate** message

Key generation



Handshake protocol



Server_key_exchange message (opt)

- The optional message **server_key_exchange** is not necessary in the following cases:
  - **Fixed Diffie-Hellmann**, **RSA encryption**
  - **pubK** is in the **certificate** message
- In contrast, it is necessary in the following cases:
  - **Anonymous DH -** p, g, Y_{svr}
  - **Ephemeral DH -** p, g, Y_{svr}, $S_{svr}(p||g||Y_{svr})$
  - **RSA (dig sig only)** - $tempPubK_{svr}$, $S_{svr}(tempPubK_{svr})$

certificate_request message

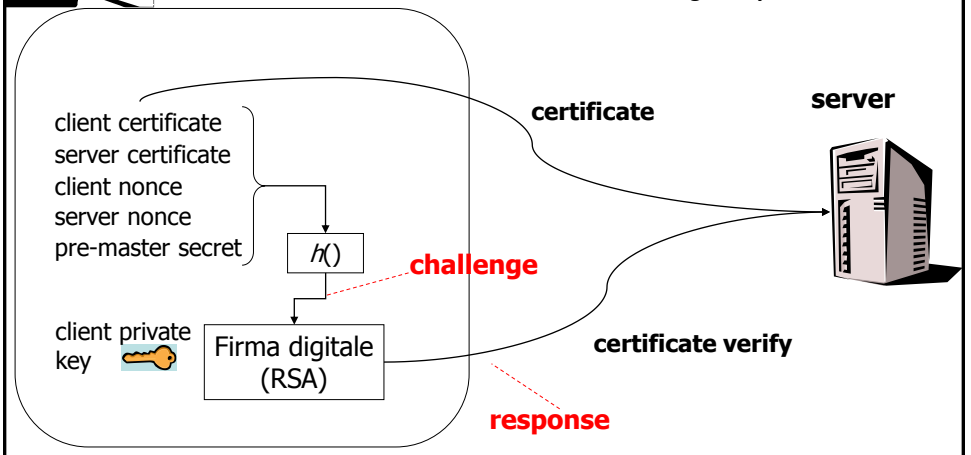
- Server may issue a **certificate_request** unless anonymous Diffie-Hellmann is used
- The message has two parameters
  - **Certificate_type**: type of digital signature and its use
    - (RSA | DSS) + (only signature | fixed Diffie-Hellmann | Ephemeral DH)
  - **Certificate_authorities:** acceptable certification authorities

Client authentication

- $\mathcal{KP}$ authenticates the server by default
- How can the client be authenticated?
  - Typically, the client is authenticated at the application level
    - password, credit card number (!!),…
- However, $\mathcal{KP}$ also supports client authentication w.r.t. the server

Client authentication

- Server requires client authentication by means of the **certificate request** msg after **server_hello**
- Authentication is based on ***challenge-response***



SSL 09/05/2018 30

Certificate & certificate_verify message

- Client sends the **certificate** message if server requested it
  - **No_certificate** alert if required certificate is not available
- The client sends **certificate_verify** message to provide explicit proof of **signing privK** possession

SSL 09/05/2018 31

Security

- Handshake Protocol
  - Nonces in client hello and server hello
    - Nonces make it possible generate a fresh master secret and avoid replay attacks
  - Certificates
    - Avoid MIM
  - Random quantities
    - Pre-master secret and nonces must be impredictable
- Record Protocol
  - A block is numbered, authenticated and encrypted
  - Avoid block replay, reordering and substitution
  - Cipher “protects” the MAC

SSL: Pros and Cons

- Pros
  - SSL is a well-designed, robust and secure protocol
- Cons
  - SSL protects communication only
  - User has to check security parameters
  - SSL is vulnerable to name spoofing

SSL

HISTORY: PITFALLS AND ATTACKS

Random generator in SSL v2.0

(on the importance of a good SPRBG)

- Pseudo-Random Bit Generator
  - keystream = H(tod || pid || ppid)
  - tod = time of day
  - pid = process id
  - ppid = parent process id
- Entropy of the triple is 47-bit => seed can be guessed in 25 s
- A more sophisticated attack based on system observation may be even more effective

Most famous attacks

- **Browser Exploit Against SSL/TLS (BEAST) attack**
  - **Weakness of CBC in TLS 1.0 (2011)**
- **Compression Ratio Info-leak Made Easy (CRIME)**
  - **Side-channel attack based on the compressed size of HTTP request (2012)**
- **Lucky13 attack**
  - **Timing side-channel attack with CBC (2013)**
- **Heartbleed attack**
  - **Buffer over-read attack (2014)**

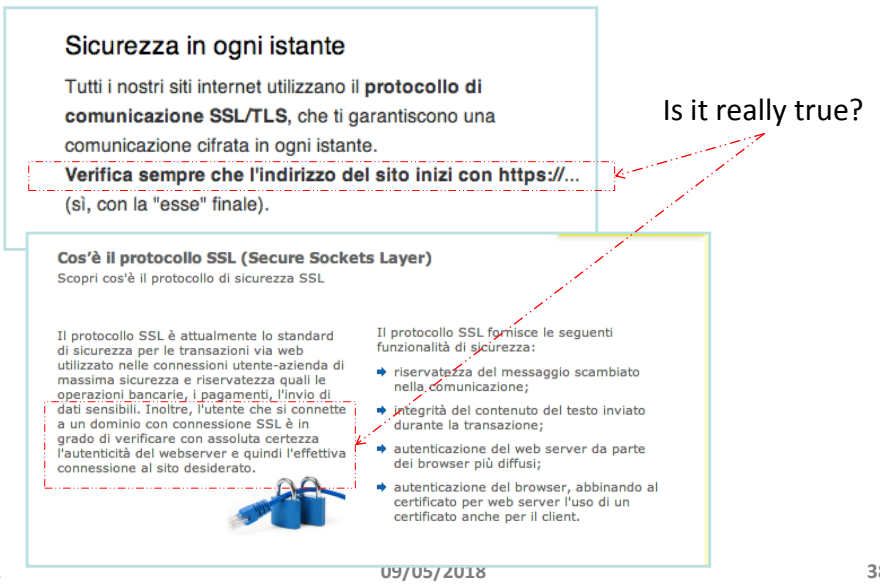
SSL 09/05/2018 36

SSL

ON USING SSL IN E-COMMERCE

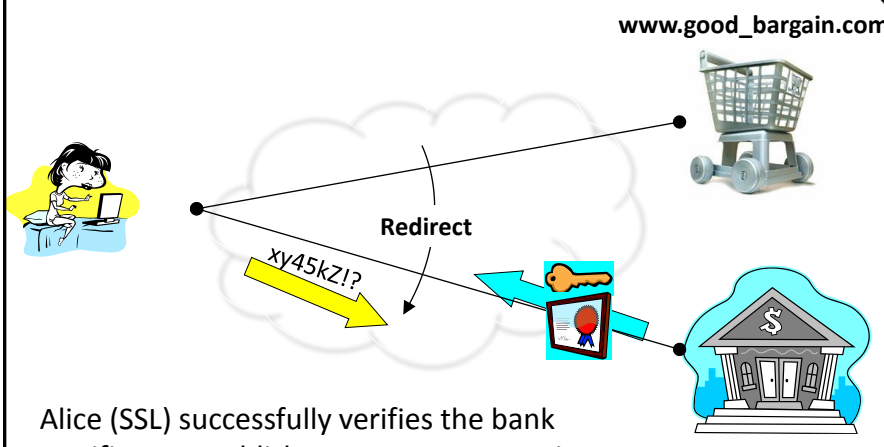
SSL 09/05/2018 37

SSL in action



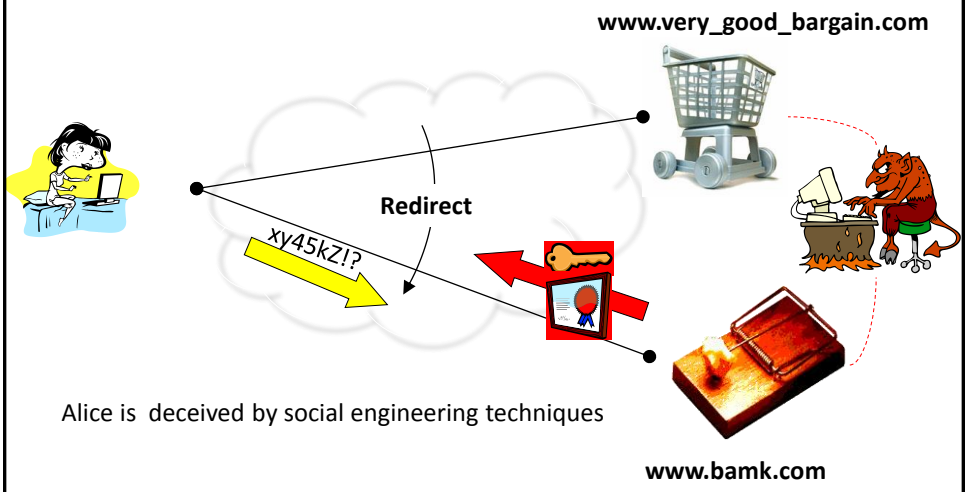
Is it really true?

MIM with SSL (1/2)



Alice (SSL) successfully verifies the bank certificate, establishes a secure connection, and sends her pwd/PIN along the connection

MIM with SSL (2/2)



Alice is deceived by social engineering techniques

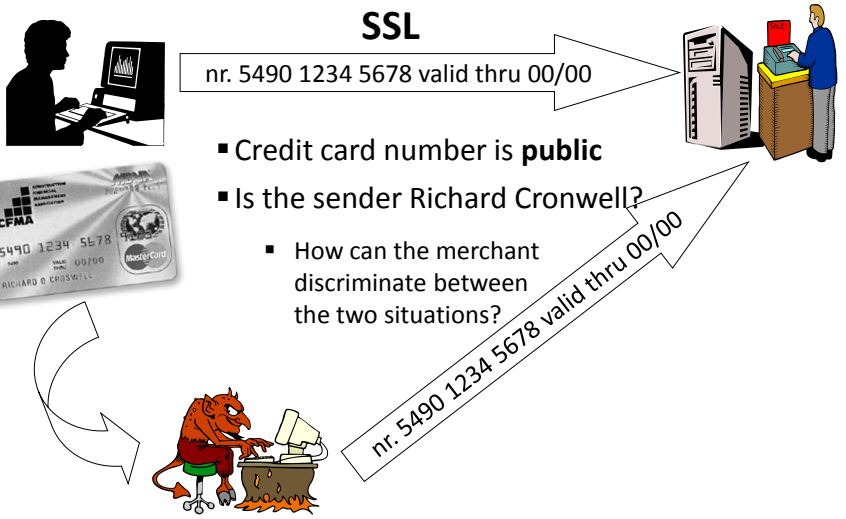
Is it the right certificate?

- **SSL operates at the transport level rather than the application level**
- **Browser controls**
  - Browser warns user if the URL known to the browser is not equal to that in the certificate (**mismatch**)
  - Browser warns user whether a certificate is signed by an unknown CA (**self-signed certificates**)
  - The user has the last word
    - The **clickthrough** phenomenon
      - Does the user understand security?
      - Usability vs security
  - These controls may be not sufficient for all web applications
  - Browser have a largely variable behaviour in this respect(*)
    - What to warn; when to warn

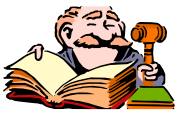
Risk allocation

- PIN/PWD is a shared secret
- In a home banking contract, the user commits himself to protect the PIN/PWD confidentiality
- In a fraud it is evident that the PIN/PWD confidentiality has been violated
- Who is liable for?

E-payment by credit card



E-payment by Credit Card



Decreto legislativo 22 maggio 1999, n. 185, di attuazione della direttiva 97/7/CE

Art. 8 - Pagamento mediante carta

1. Il consumatore può effettuare il pagamento mediante carta ove ciò sia previsto tra le modalità di pagamento, da comunicare al consumatore al sensi dell'articolo 3, comma 1, lettera e), del presente decreto legislativo.

2. L'istituto di emissione della carta di pagamento riaccredita al consumatore i pagamenti dei quali questi dimostri l'eccedenza rispetto al prezzo pattuito ovvero l'effettuazione mediante l'uso fraudolento della propria carta di pagamento da parte del fornitore o di un terzo, fatta salva l'applicazione dell'articolo 12 del decreto-legge 3 maggio 1991, n. 143, convertito, con modificazioni, dalla legge 5 luglio 1991, n. 197. L'istituto di emissione della carta di pagamento ha diritto di addebitare al fornitore le somme riaccreditate al consumatore.

E-payment by Credit Card

- **Gli istituti di emissione**, cui compete l'autorizzazione dell'operazione di pagamento, nonché i soggetti che rendono tecnicamente possibile la transazione on-line, **sono tenuti a controllare la correttezza del numero della carta e la data della sua scadenza ma non anche la corrispondenza tra il numero fornito e l'effettivo titolare**
- Gli istituti di emissione verificano la corrispondenza tra numero della carta di credito comunicato per effettuare una transazione on-line ed il nominativo fornito da colui che la effettua.

  Ad esempio, l'**Address Verification Service (AVS)** verifica che l'indirizzo di consegna sia quello con cui il possessore della carta è registrato
- In Europa il grado di sicurezza nelle transazioni on-line è minore e quindi il commercio elettronico è destinato ad incontrare resistenze anche da parte dei fornitori di che sopportano rischi elevati

E-payment by Credit Card: risk allocation

- Il fornitore di beni o servizi on-line **è tenuto ad accollarsi il rischio** della rivalsa degli istituti di emissione qualora, in caso di uso fraudolento delle carta, questi riaccreditano le corrispondenti somme al legittimo titolare.
- La legge **non consente** al fornitore di liberarsi dall'obbligo della restituzione delle somme agli istituti di emissione qualora dimostri
  1. di avere usato tutte le cautele necessarie e possibili ad evitare l'uso fraudolento della carta di credito
  2. che il fatto è stato causato dal caso fortuito.
- I fornitori dovranno usare tutte le cautele del caso per potere, nel caso di uso fraudolento di carte di credito, perlomeno rintracciare l'illegittimo utilizzatore e rivalersi su questo.

  Le conseguenze derivanti dall'addebito delle somme riaccreditate al titolare della carta potrebbero poi essere annullate contraendo una **assicurazione** a copertura dei danni (economici) derivanti da tale circostanza.

E-payment by Credit Card

Foglio informativo sulle operazioni e servizi offerti alla clientela (CariPrato)

Caratteristiche e rischi tipici

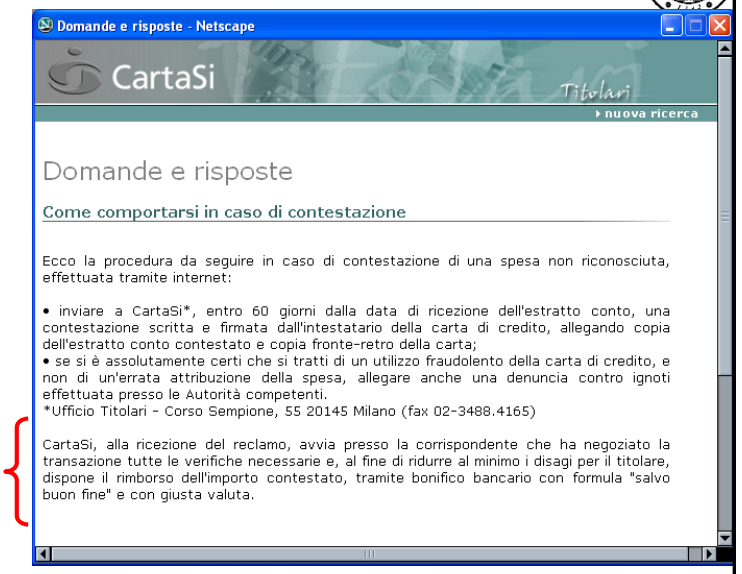
Struttura e funzione economica
CARTE DI DEBITO e CARTE DI CREDITO

Strumenti di pagamento rilasciabili a clienti della Banca che consentono:
- Acquisto di beni;
- Prestazione di servizio presso esercenti convenzionati.
- Ottenimento di contante presso sistemi automatici o sportelli bancari convenzionati.

Funzione Bancomat: è il servizio in forza del quale la banca (emittente), attraverso il rilascio di una Carta, consente al correntista (c.d. "titolare") di effettuare prelievi di denaro — entro massimali di utilizzo stabiliti dal contratto - presso sportelli automatici (ATM) contraddistinti dal marchio Bancomat, digitando un codice segreto (c.d. P.I.N., "Personal Identification Number").
Funzione PagoBANCOMAT: è il servizio in forza del quale il correntista può compiere acquisti di beni e servizi presso esercizi commerciali convenzionati che espongono il marchio "PagoBANCOMAT", digitando il citato codice segreto.
L'utilizzo del sistema di pagamento è consentito nei limiti giornaliero e mensile, entro limiti di importo contrattualmente previsti, determinato dal momento dell'emissione e dalla capienza di conto corrente al momento dell'addebito.

Principali rischi (generici e specifici)
Il rischio relativo ad eventuali utilizzi fraudolenti effettuati con le Carte di Pagamento è limitato a 150 € per evento se il Titolare ha ottemperato e rispettato quanto indicato dalla "Raccomandazione della Commissione Europea del 30 giugno 1997 n. 97/489"
In sintesi il titolare è tenuto a:
- Firmare la carta nel caso che la stessa sia munita di apposita banda di scrittura;
- Osservare la massima attenzione nella custodia della carta e PIN e la massima riservatezza nell'uso del medesimo;
- Bloccare la carta nel caso di furto, smarrimento o uso fraudolento della medesima, confermando l'evento con denuncia o dichiarazione di smarrimento.

E-payment by Credit Card

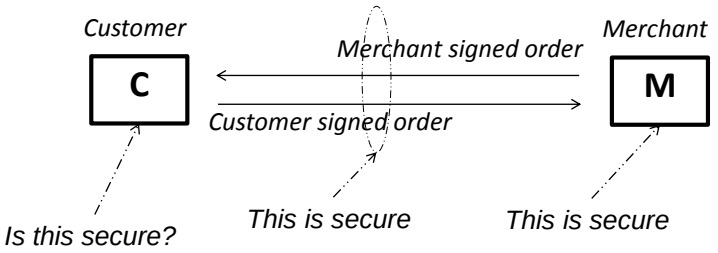


Secure Electronic Transactions

- SET was built to answer to these problems
- SET has been designed and implemented in the late 90's
  - Commissioned by Visa and Mastercard
  - Involves all (IBM, Microsoft,…)
- SET was a failure
  - Too "heavy"
  - Too expensive
  - Specifications takes more than 1000 pages (!)
- We are interested in the risk allocation

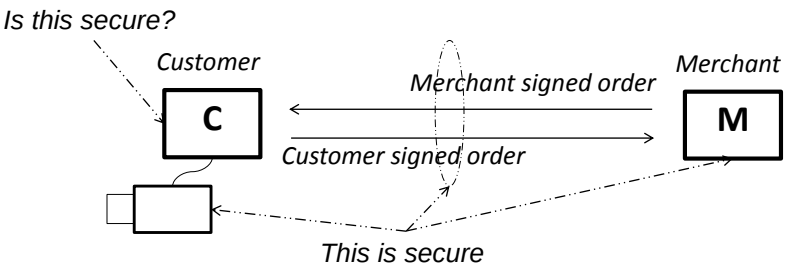
Secure Electronic Transactions

- SET requires a PKI in place
- A (privK, pubK) pair is stored at M and C
- If an order is signed by your key you cannot repudiate it
  - The risk is allocated on the customer
- M and C are assumed trusted devices!
  - Stealing a privK is equivalent to stealing a file



Secure Electronic Transactions

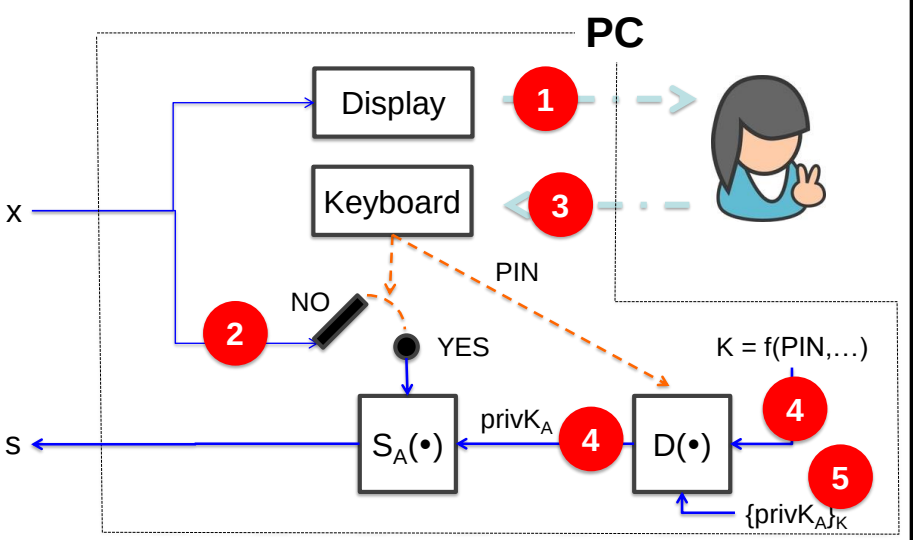
- Do smart cards help?
  - Loosing a piece of plastic vs. loosing a file
  - Is what you see what you sign?



Dancing on the cliff rim

1. The owner keeps secret the privK
2. If V(pubK, s) == True then s was made by privK
3. A valid, unrevoked certificate links pubK to a name
4. Certification process links name to owner
5. Therefore the owner has generated signature s (and he is responsible of it)

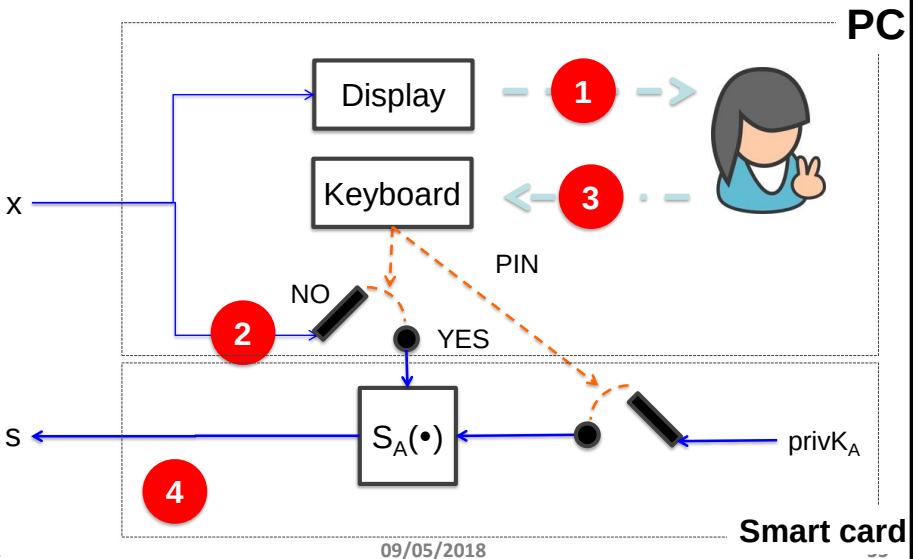
Attack surface (I)



Attack surface (II)

1. The attacker shows Alice x' instead of x
2. The attacker shows Alice x but signs x'
3. The attacker changes Alice's decision or steals Alice's PIN
4. The attacker steals the private key
5. The attacker steals the encrypted private key for an offline attack

Attack surface (III)



Attack surface (IV)

1. The attacker shows Alice x' instead of x and trick Alice into signing x
2. The attacker replaces x with x'
3. The attacker changes Alice's decision or steals Alice's PIN
4. The attacker steals and attacks the smart card
   1. Physical attack
   2. Side-channel attack