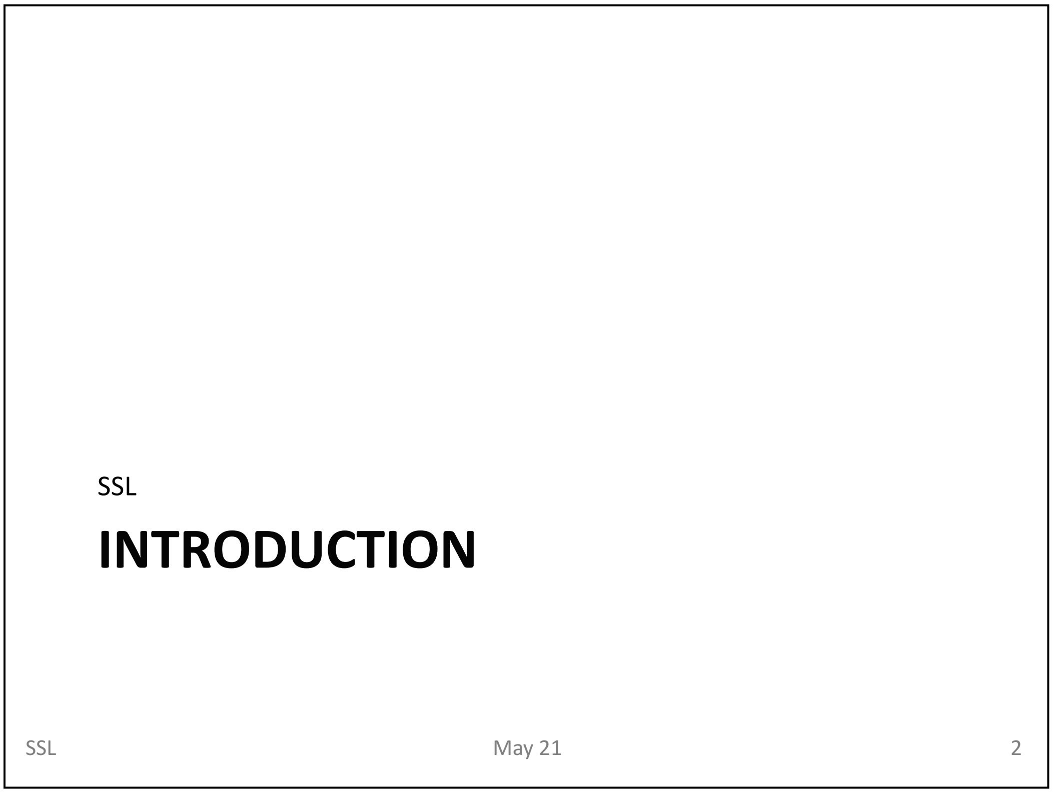
Secure Socket Layer (SSL)

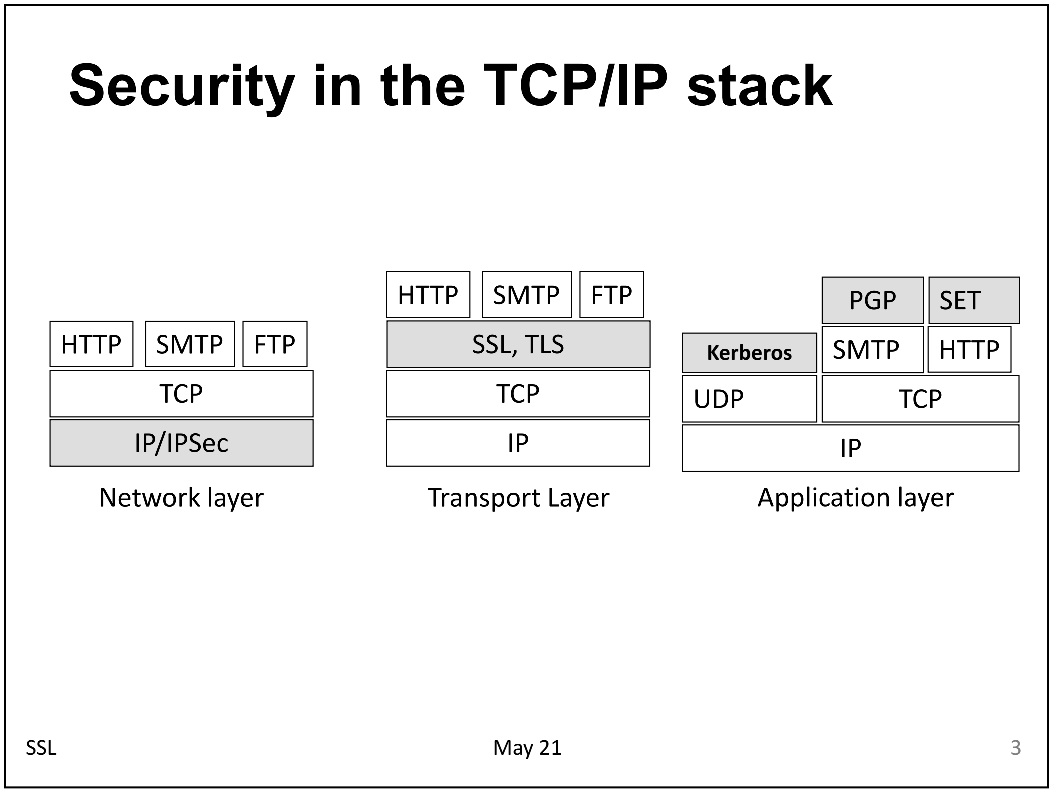
Gianluca Dini
Dept. of Ingegneria dell’Informazione
University of Pisa
Email: gianluca.dini@unipi.it

Version: 2021-05-10



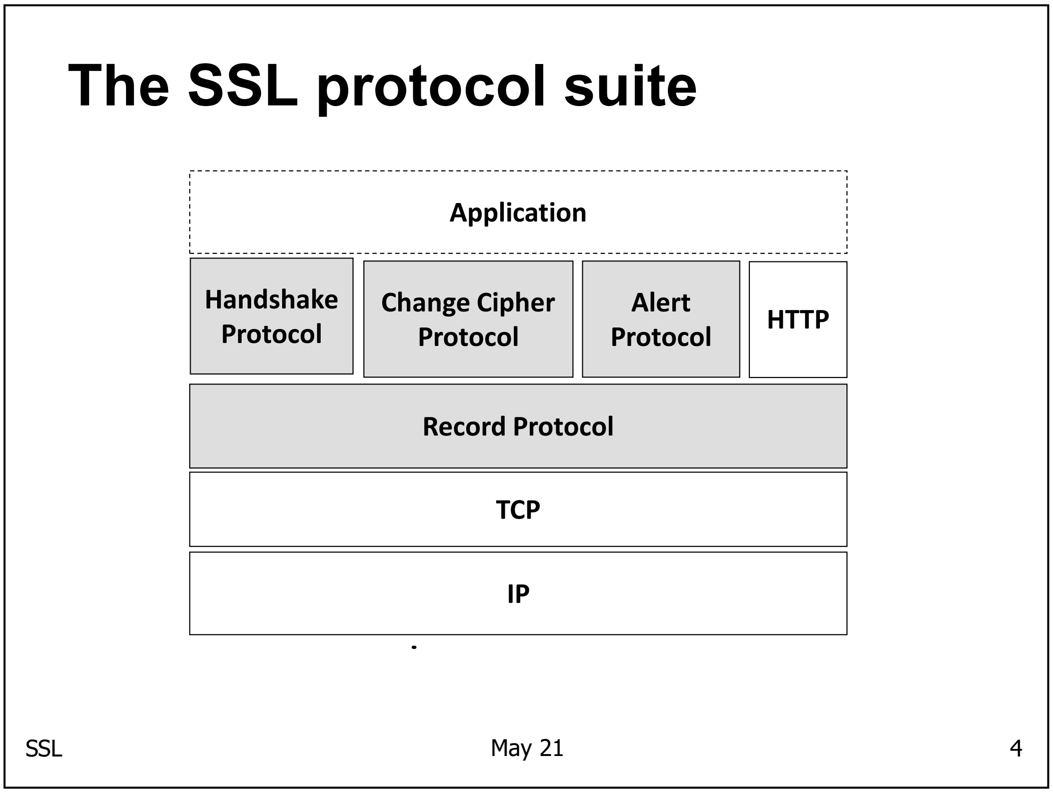
SSL

INTRODUCTION



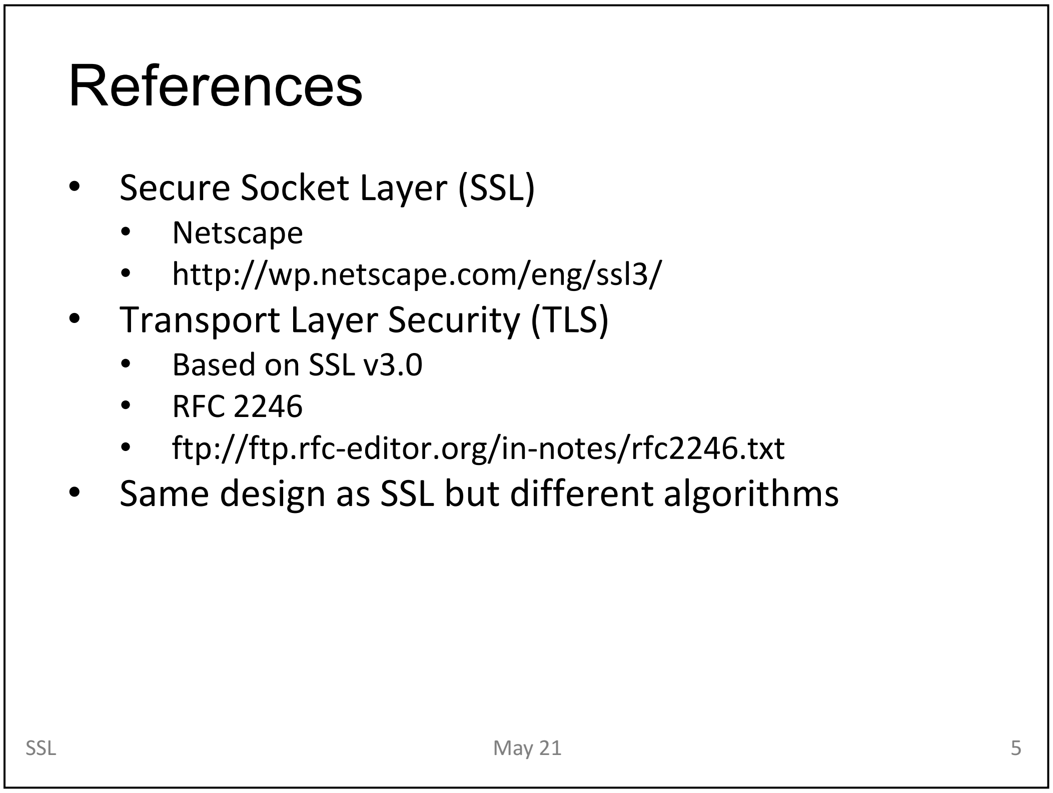
Security in the TCP/IP stack

SSL May 21 3



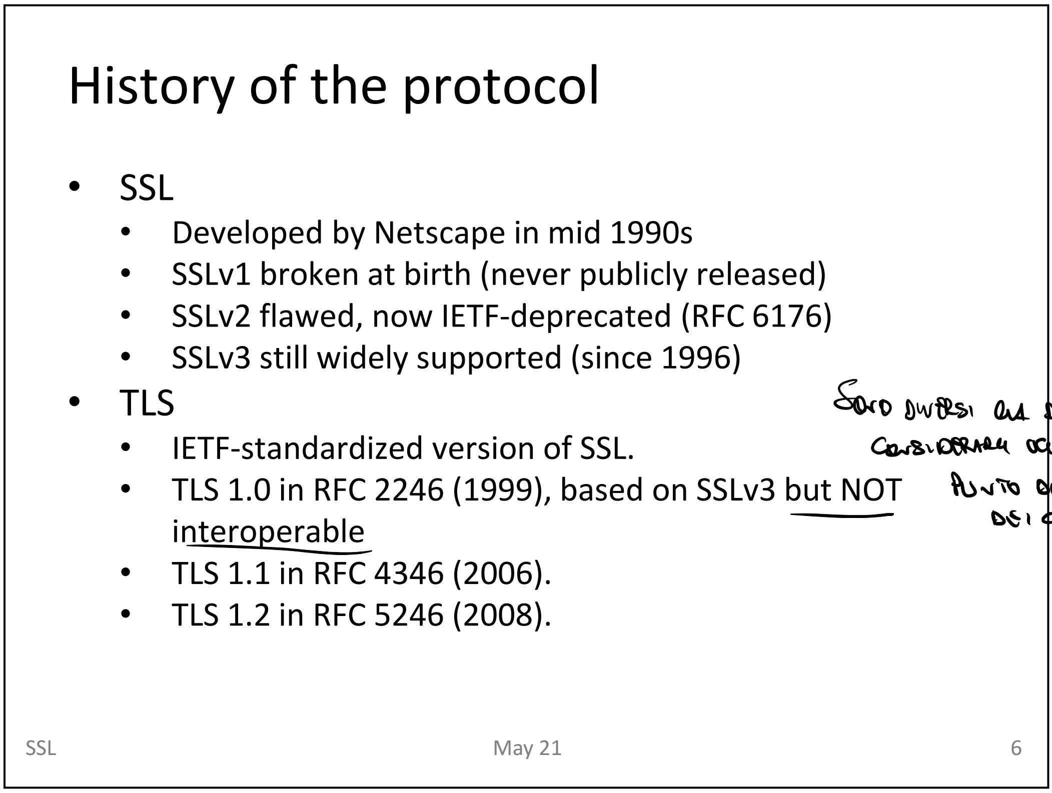
The SSL protocol suite

SSL May 21 4



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).

SSL May 21 6

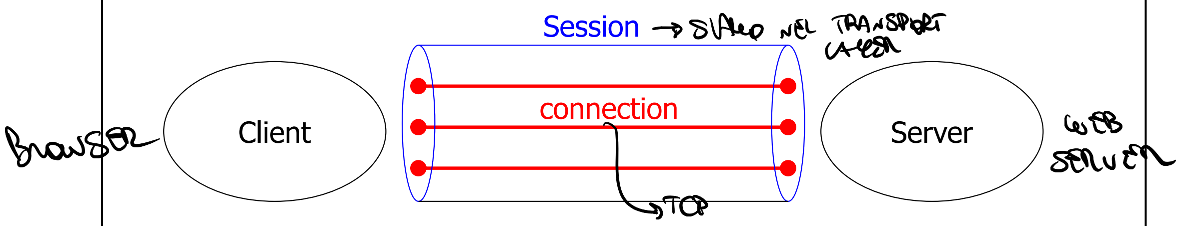
Sono diversi ma possiamo considerarli uguali dal punto di vista dei concetti

TLS is very similar to SSLv3, however differences exist

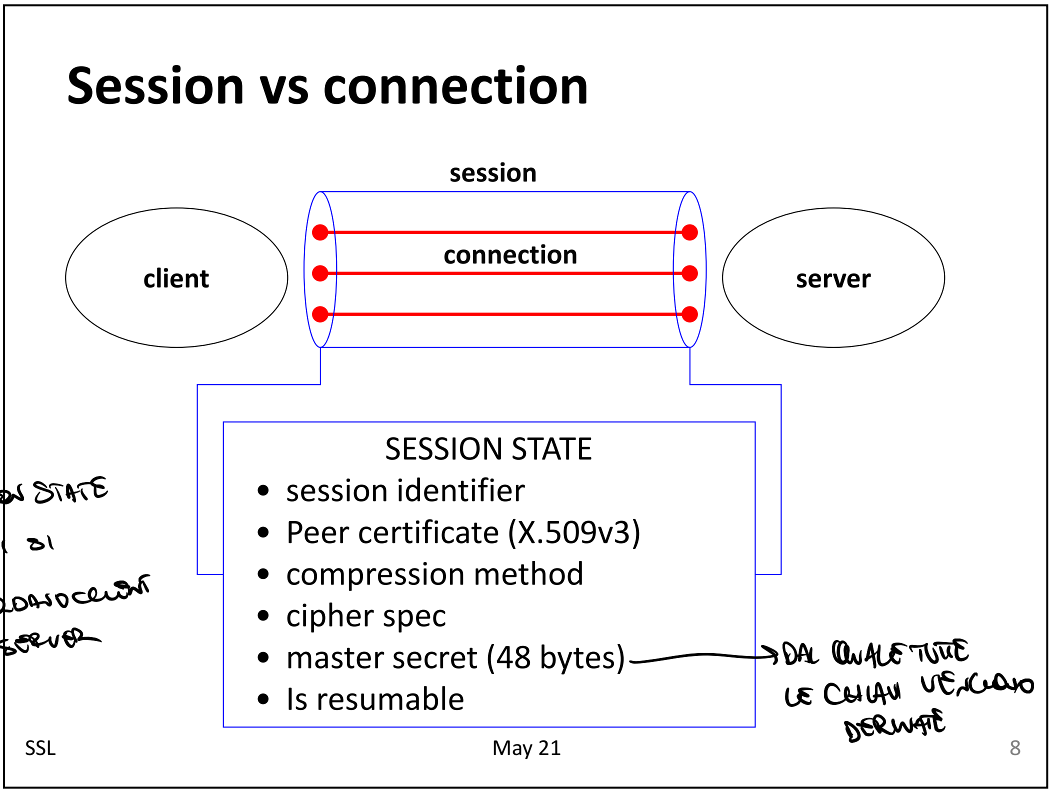
- Minor version number = 1
- Slight difference in computing the MAC
- TLS uses a PRF in order to expand blocks for generation or validation of keys. SSLv3 uses MD5.
- Alarm codes – TLS supports the alarm codes as SSLv3 but no_certificate. However, TLS introduces additional codes, both *fatal* and not
- Cipher suite
- Client certificate type
- Certificate_verify and finished messages
- Padding size: the minimum in SSLv3; arbitrary in TLS in order to discourage traffic analysis

SSLv3 and TLS cannot interoperate.

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 → PUBLIC KEY ENCRYPTION
  - Avoid **expensive** negotiation of crypto pars for each connection



Session vs connection

SESSION STATE

- session identifier
- Peer certificate (X.509v3)
- compression method
- cipher spec
- master secret (48 bytes) → DAL QUALE TUTTE LE CHIAVI VENGONO DERIVATE
- Is resumable

COMMON STATE SU CUI SI ACCORDANO CLIENT E SERVER

SSL May 21 8

The is resumable field specifies whether the session can be used to establish new connections.

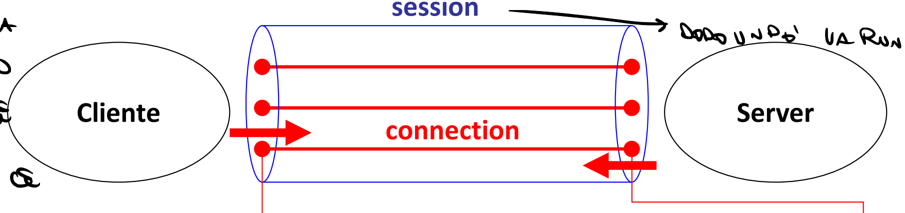
ISTANZIARE LA SESSIONE RICHIEDE MOLTE COMPUTAZIONI → PUBLIC KEY

Session vs connection

UNA SESSIONE E' COMPOSTA DA PIU' CONNESSIONI

DOPO UN PO' VA RINNOVATA

QUANDO SI AVVIA LA CONNESSIONE NON VOGLIO RIPETERE QUINDI SFRUTTO LE INFORMAZIONI RICAVATE PRIMA → NON RICHIEDONO PUBLIC KEY



DA COSA E' COMPOSTA UNA CONNESSIONE

CONNECTION STATE

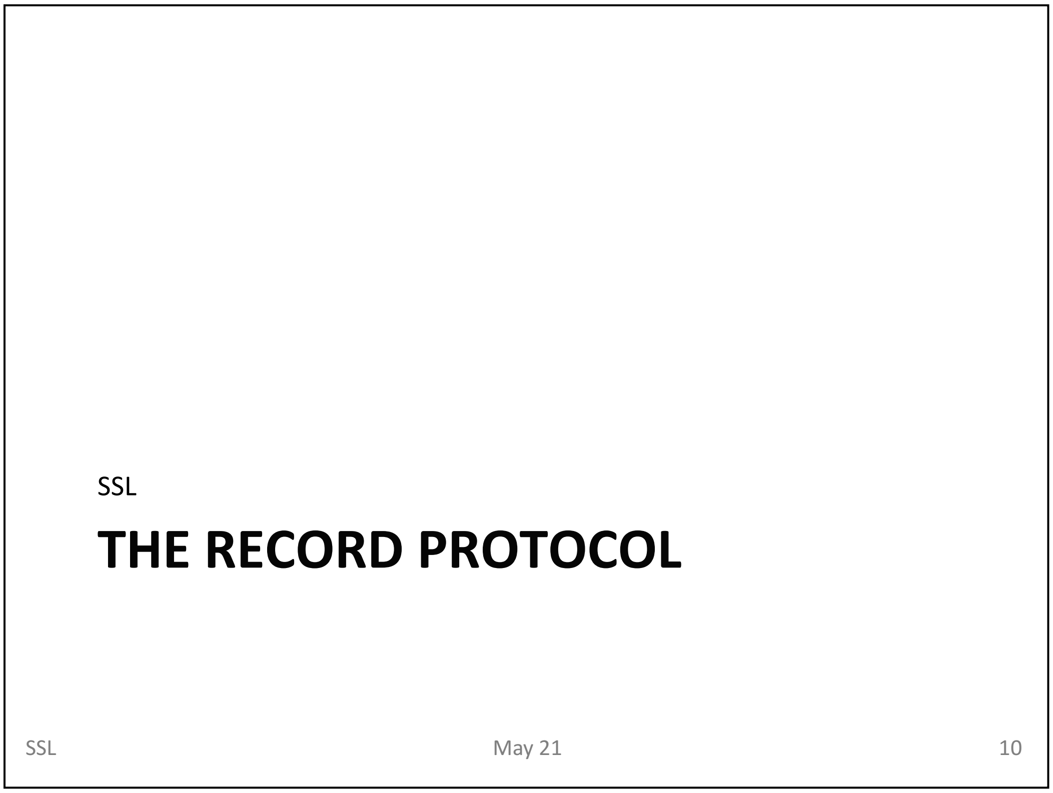
- Server random number (nonce)
- Client random number (nonce)
- Server write MAC secret
- Client write MAC secret
- Server write key
- Client write key
- Initialization vectors
- Sequence numbers

Server/Client write MAC secret: GENERATO DA MASTER SECRET → AUTHENTICITY

Server/Client write key: GENERATE A PARTIRE DA MASTER SECRET → CONFIDENTIALITY

SSL May 21 9

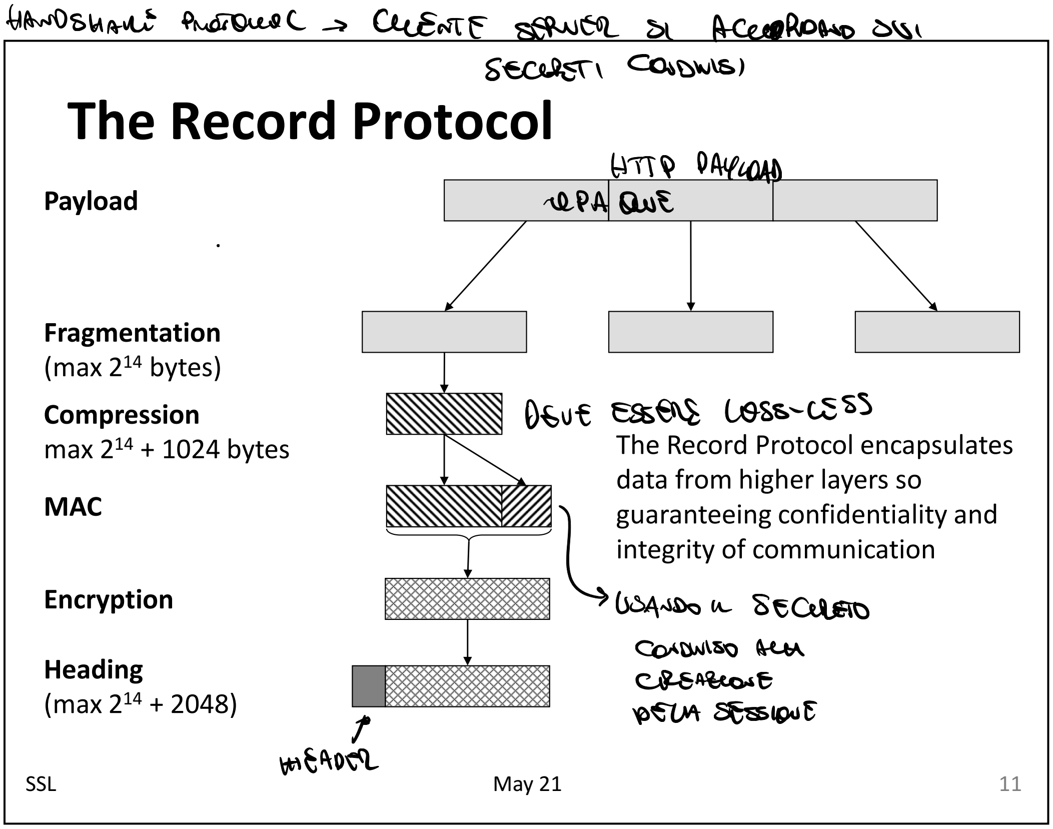
Each connection is associated to a session. There may be several connections for a given session. Typically, at a given time, there exists only one session between two parties.



SSL

THE RECORD PROTOCOL

HANDSHAKE PROTOCOL → CLIENTE SERVER SI ACCORDANO SUI SEGRETI CONDIVISI

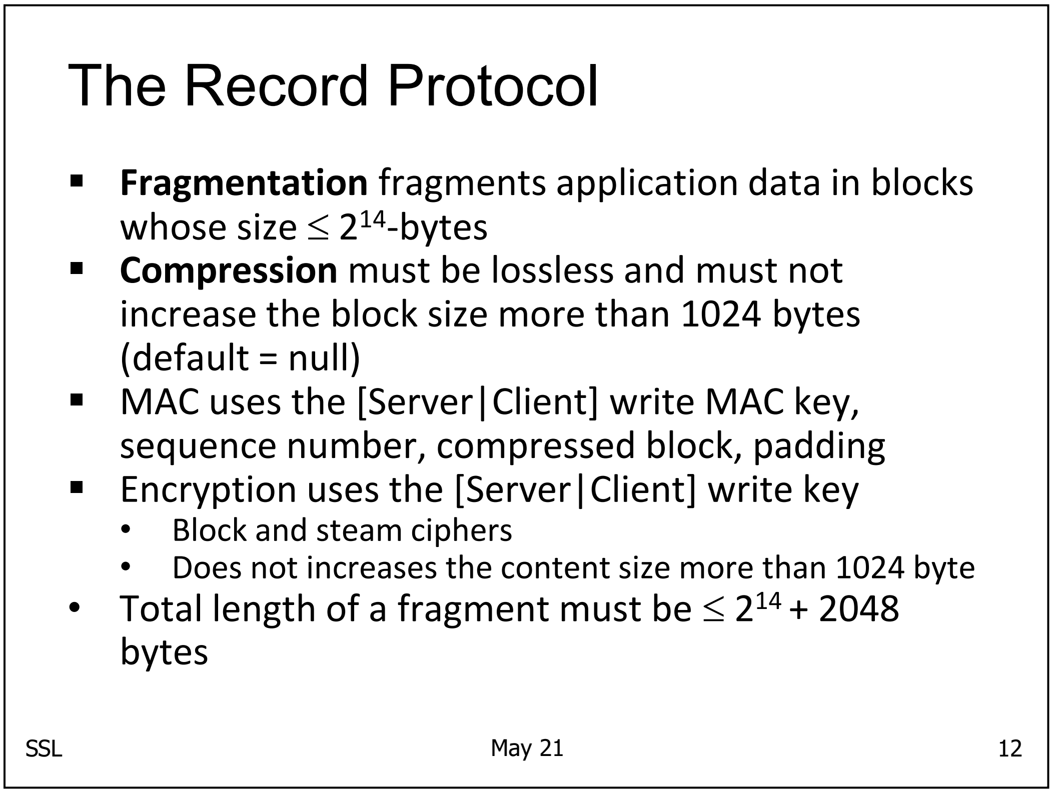


The Record Protocol

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

SSL May 21 11

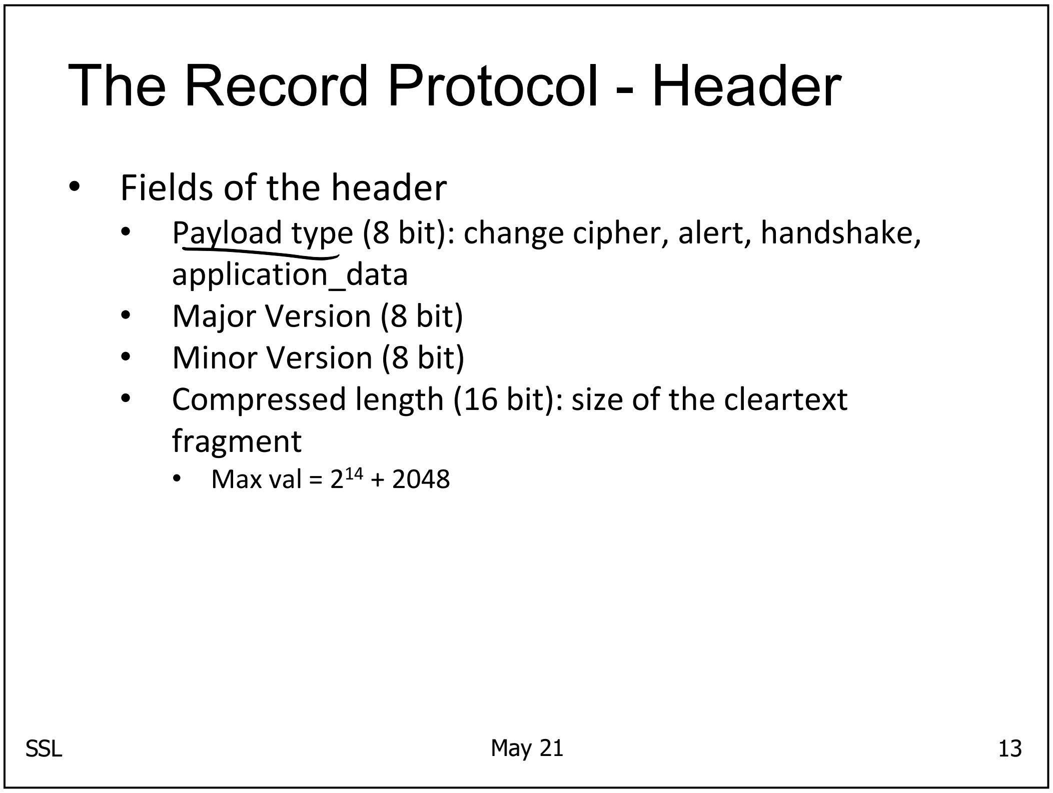
The Record Protocol provides confidentiality and integrity to SSL connections. Key for MAC and encryption are established by the Handshake protocol.



The Record Protocol

- **Fragmentation** fragments application data in blocks whose size $\leq 2^{14}$-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

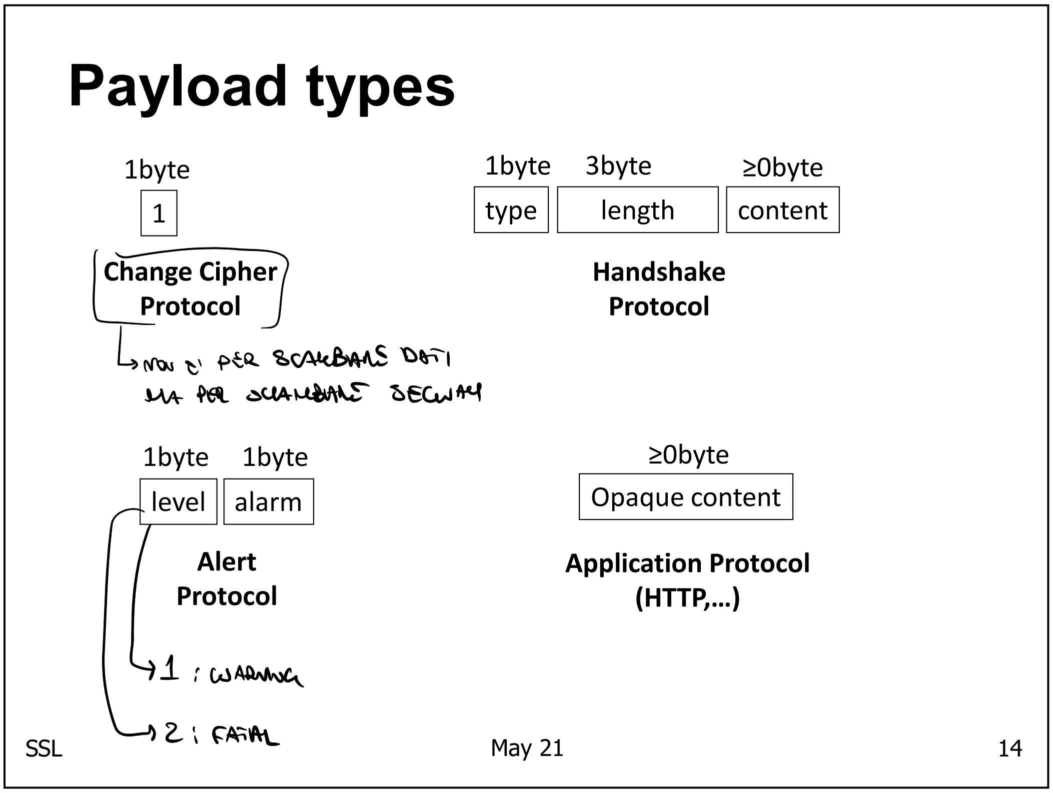
Compression should reduce the length of data. However, for small blocks, compression may expand data.
In case of stream cipher, compressed message and MAC are encrypted. In case of block cipher, padding is added after MAC, before encrypting.



The Record Protocol - Header

- Fields of the header
  - Payload type (8 bit): change cipher, alert, handshake, application_data
  - Major Version (8 bit)
  - Minor Version (8 bit)
  - Compressed length (16 bit): size of the cleartext fragment
    - Max val = 2^{14} + 2048

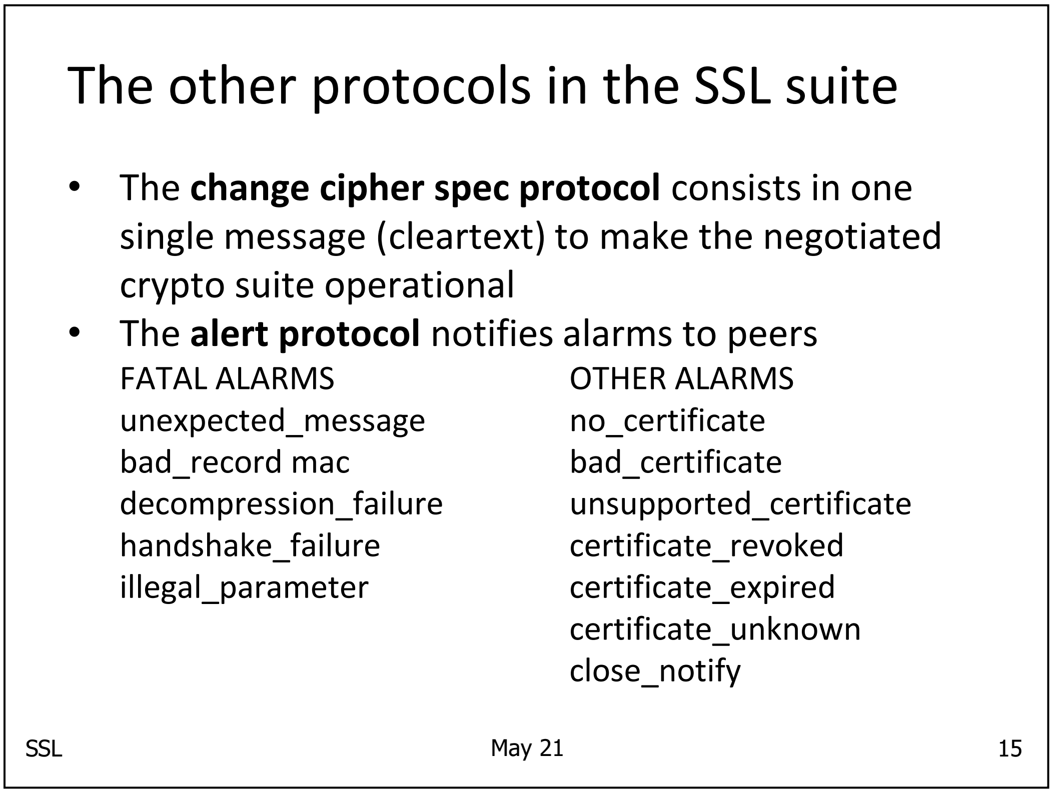
In case of SSLv3, major version is 3 and minor version is 0.
The compressed lenght specifies the size of the cleartext fragment or the compressed fragment is compression is used.



Payload types

SSL May 21 14

The *Change cipher spec* protocol consists of a single one-byte message whose value is 1.
The *Alert protocol* is used to communicate the peer alarm messages. The *level field* may assume two values: warning (1) or fatal (2), according to the severity of the message. A fatal alert terminates SSL connection and no new connection in the same session can be established. Already existing connections may continue. The *alarm field* specifies the specific alarm. Fatal alarm are: unexptected_message, bad_record_mac, decompression_failure, handshake_failure, illegal_parameter. Other alarms are: close_notify, no_certificate, bad_certificate, unsupported_certificate, certicate_revoked, certificate_expired, certificate_unknown.
The Handshake protocol is the most complex protocol in SSL. This protocol allows client and server to authenticate each other, negotiate a cryptographic suite and establish the cryptographic keys as needed. The Handshake protocol precedes any application protocol. Each message of the Handshake protocol consists of three fields: type (1byte) which specifies one of the 10 different types of messages in the Handshake protocol; length (3 bytes) that species the length of the message; contents (≥ 1 byte) which specifies the parameters associated with the message.



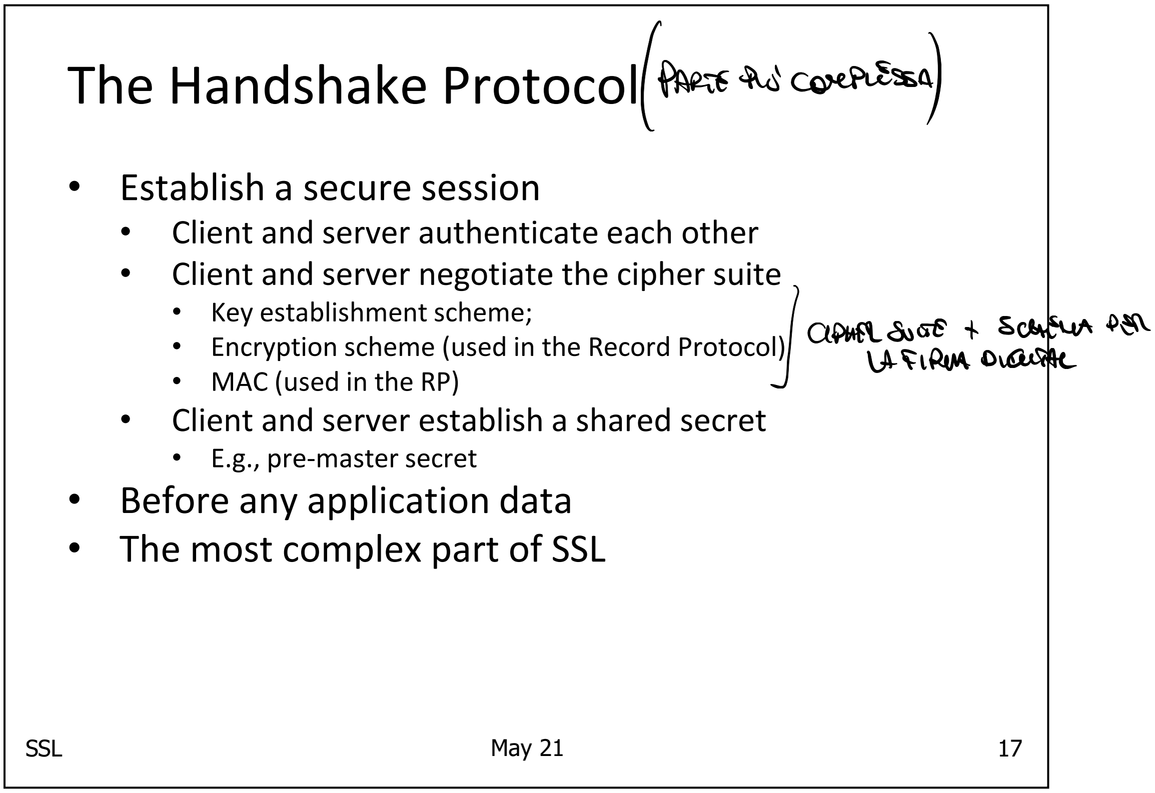
The other protocols in the SSL suite

- 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

SSL

THE HANDSHAKE PROTOCOL



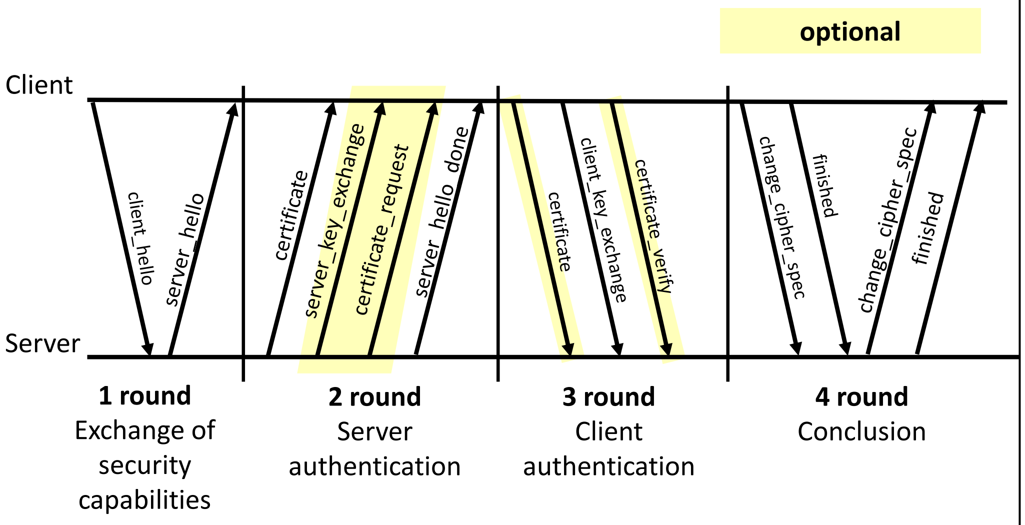
The Handshake Protocol (PARTE PIÙ COMPLESSA)

- 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 Record Protocol)
    - MAC (used in the RP)

    CIPHER SUITE + SCELTA PER LA FIRMA DIGITALE
  - Client and server establish a shared secret
    - E.g., pre-master secret
- Before any application data
- The most complex part of SSL

SSL May 21 17

Handshake protocol: an overview



1 round
Exchange of security capabilities

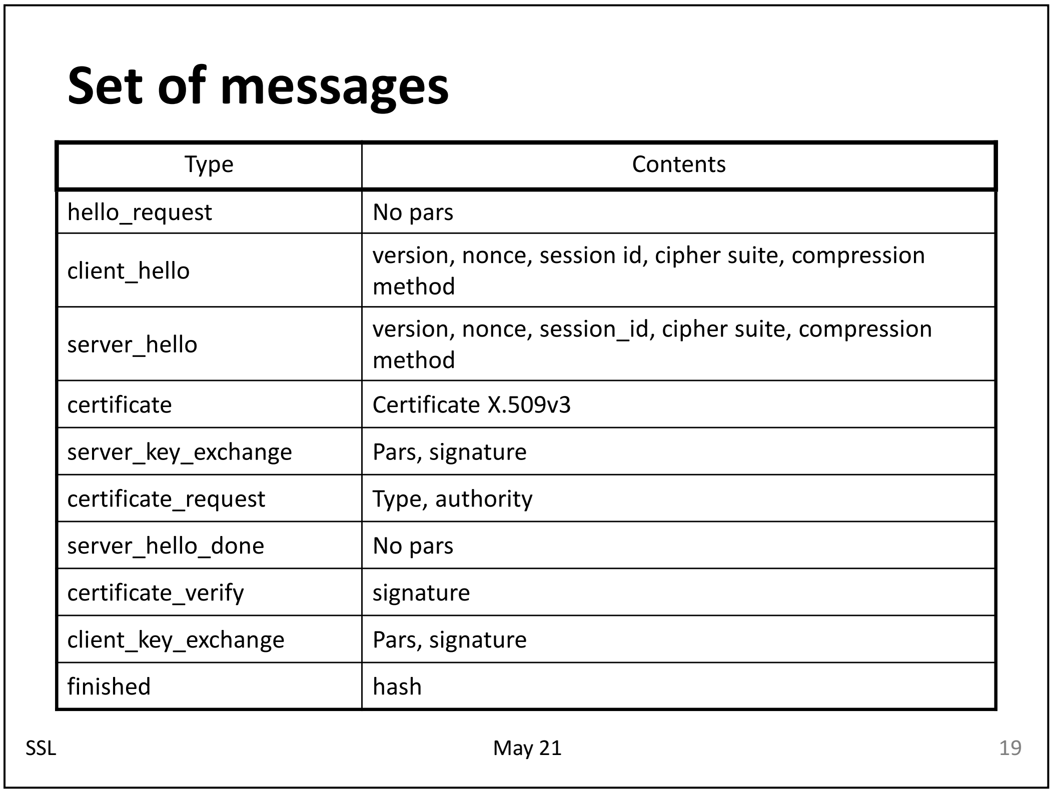
2 round
Server authentication

3 round
Client authentication

4 round
Conclusion

13 MESSAGGI

INTEROPERABILITY → RENDE PROTOCOLLI COMPLICATI!

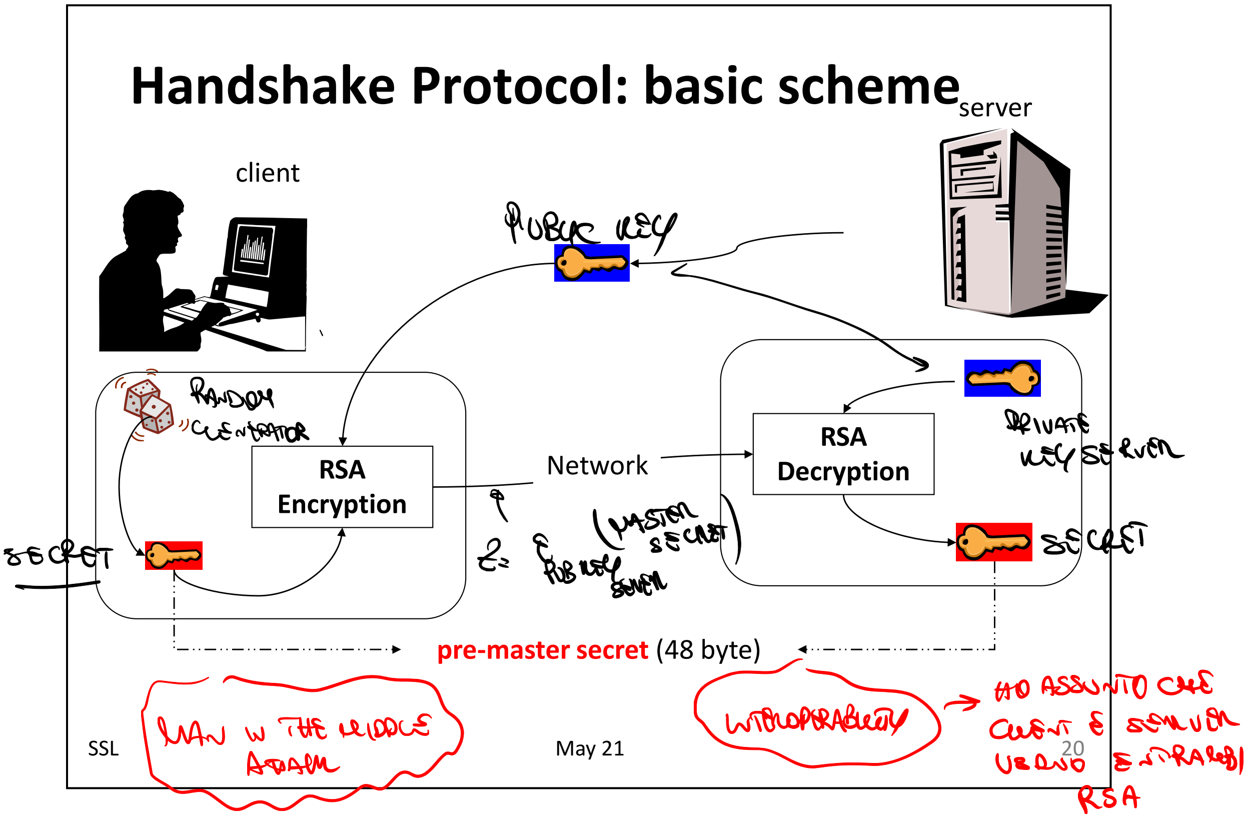


Set of messages

Type	Contents
hello_request	No pars
client_hello	version, nonce, session id, cipher 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

Fields of the client_hello/server_hello messages.

- Version: The highest SSL version the client supports
- Nonce: 32-bit timestamp || 28-bytes random
- Session ID
    - If the session_id is <> 0, then the client wants to create a new connection in the session specified by session_id (or to update a connection);
    - If the session_id is ==0, then the client wants to create a new connection in a new session
- Cipher suite: the list of combinations of crypto algs desired by the client in descending order of priority
- Compression method: the list of compression methods supported by the client



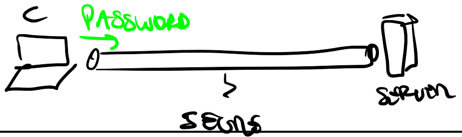
Handshake Protocol: basic scheme

pre-master secret (48 byte)

MAN IN THE MIDDLE ATTACK

INTEROPERABILITY → HO ASSUNTO CHE CLIENT E SERVER USANO ENTRAMBI RSA

SSL May 21 20

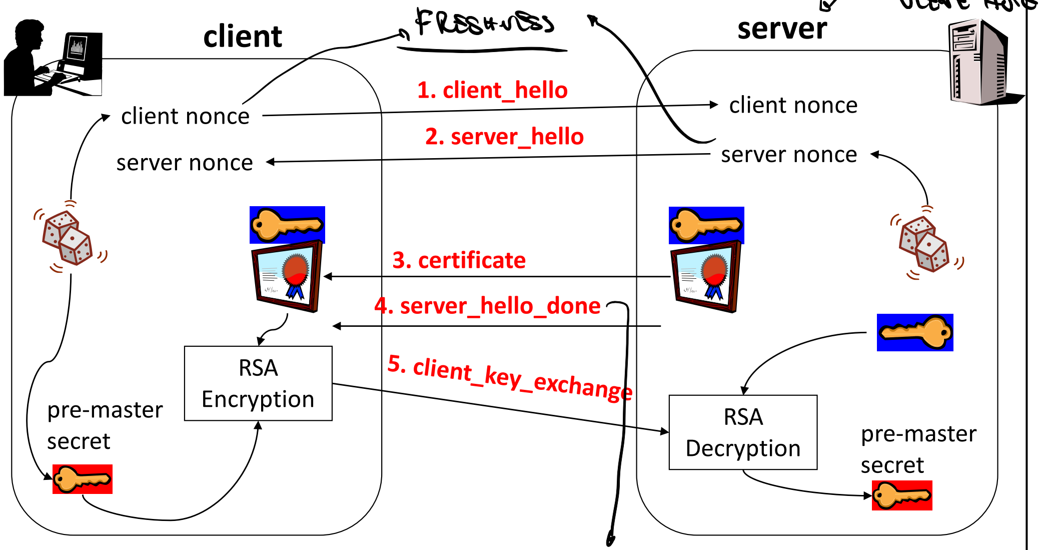


IL CLIENT VIENE AUTENTICATO AD APPLICATION LAYER

Handshake Procol: basic scheme

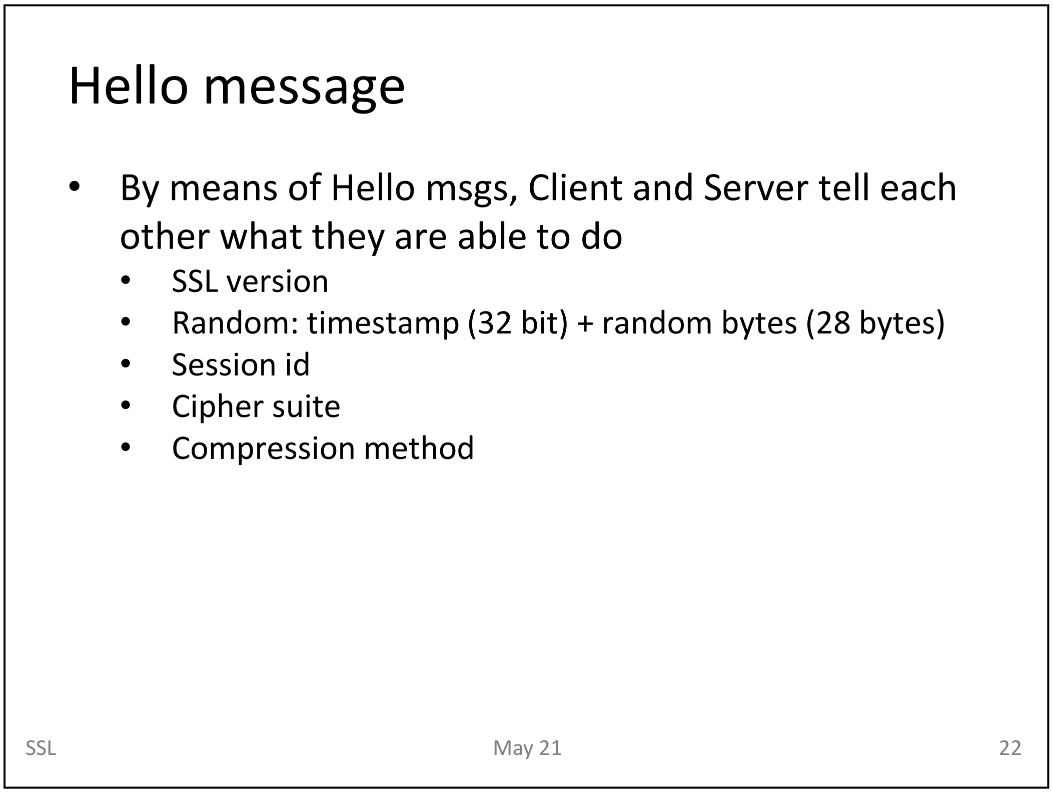
Server Authentication only

SOLO IL SERVER VIENE AUTENTICATO



SSL May 21 21

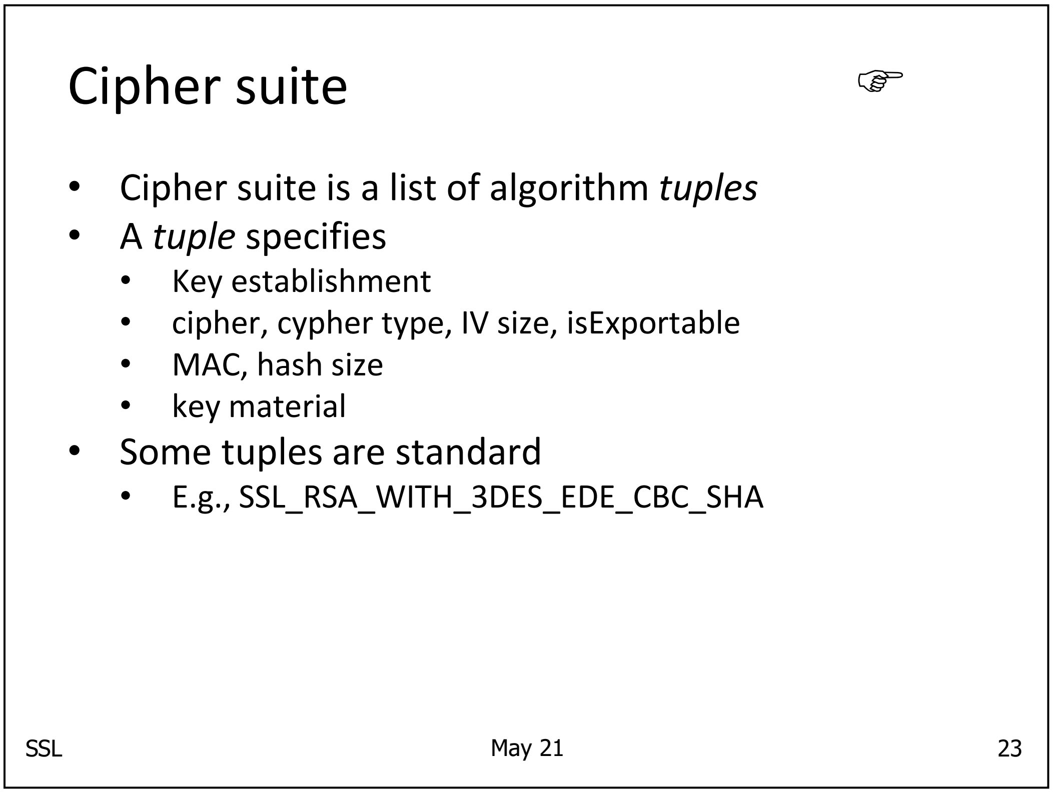
IL SERVER POTREBBE RICHIEDERE UN CERTIFICATO AL CLIENT OPPURE NO, SERVE PERCHE' GARANTISCE CHE IL ROUND 2 E' FINITO.



Hello message

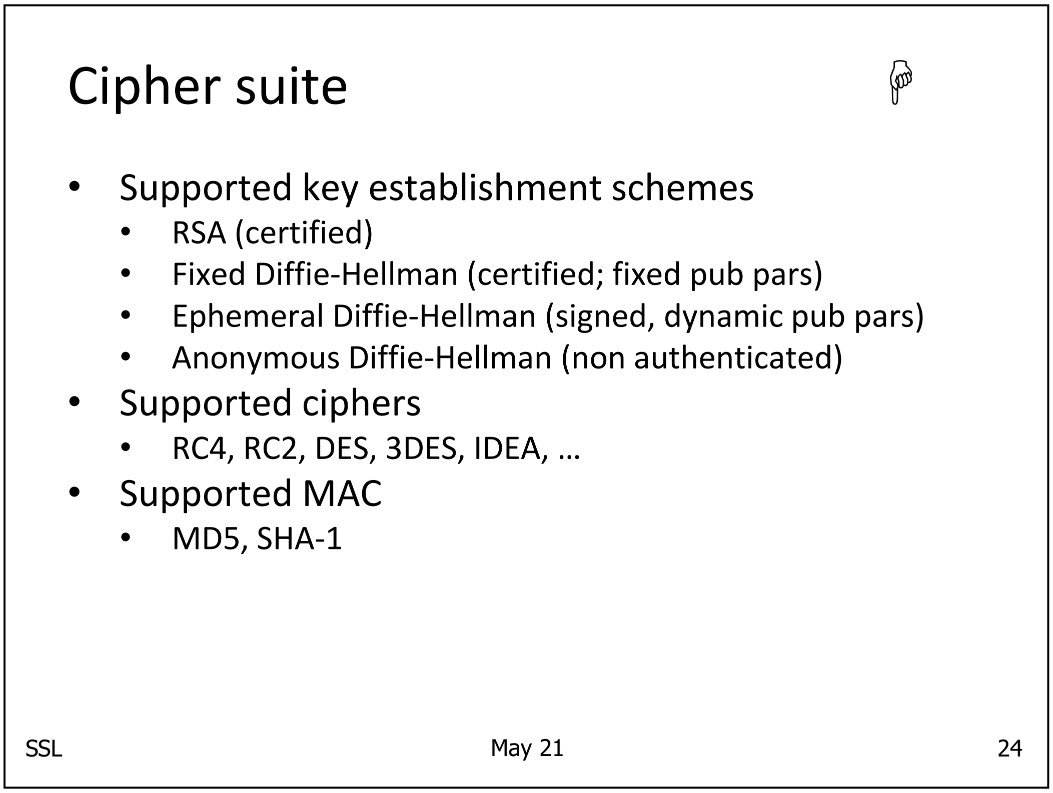
- By means of Hello msgs, Client and Server tell each other what they are able to do
  - SSL version
  - Random: timestamp (32 bit) + random bytes (28 bytes)
  - Session id
  - Cipher suite
  - Compression method

SESSION ID. A session id different of zero denotes that the client wishes to update the parameters of an existing connection or create a new connection in the current session. A session id equal to zero specifies that the client requires to instaurate a new connection in a new session.
CIPHER SUITE. The client specifies a list of supported cryptographic algorithms. Each element of the lis specifies the key establishment algorithm, the encryption algorithm and the MAC algorithm. Elements are sorted in order of decreasing preference. The server chooses one of them
COMPRESSION METHOD. It specifies the list of compression algorithms in order of decreasing preference.



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

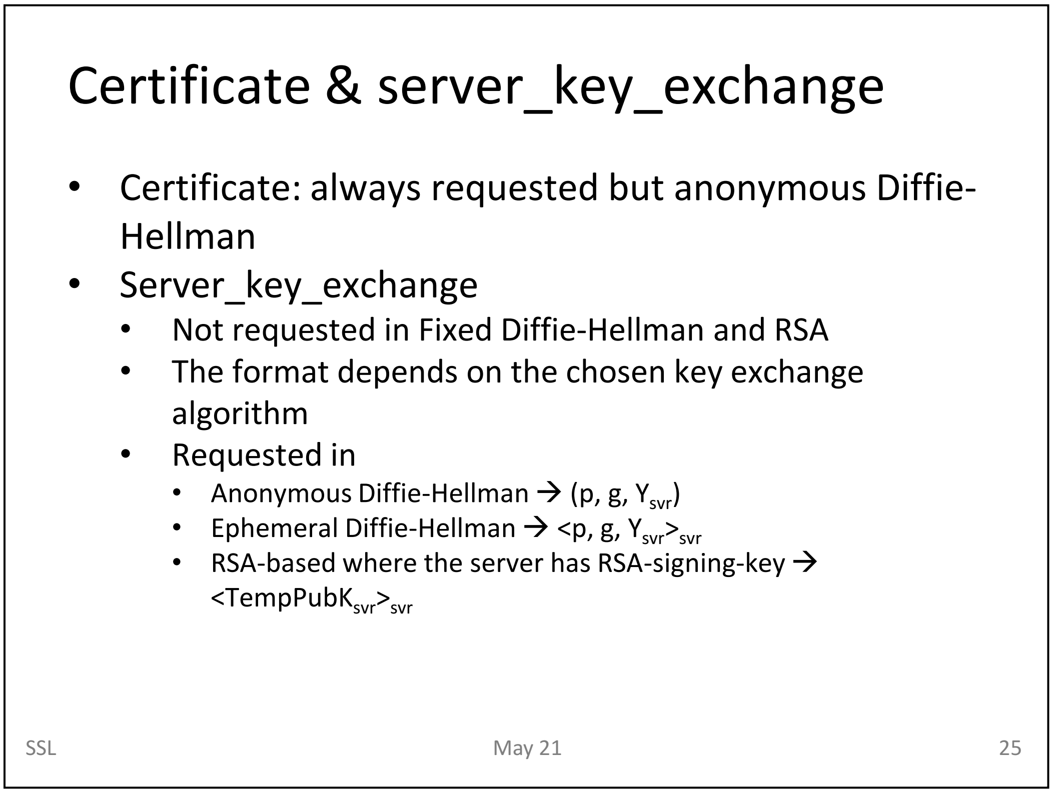


Cipher suite

- 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

SSL May 21 24

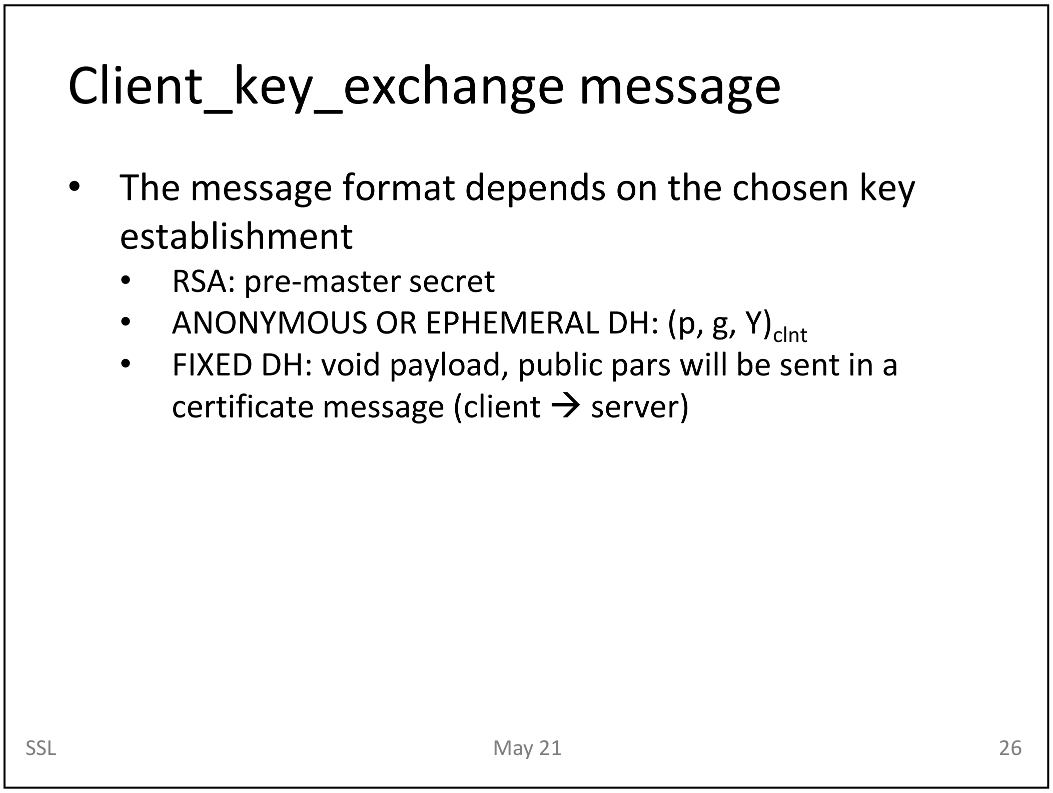
- RSA. Il pre-master secret is encrypted with the recipient’s public key which is certified.
- FIXED DH. Server’s public parameters (p, g, Y_{svr}) are certified by a Certification Authority (CA). Client’s certificate is required if client authentication is required.
- EPHEMERAL DH. This protocol guarantees Perfect Forward Security. DH public keys are digitally signed (RSA or DSS).Signing keys are certified.
- ANONYMOUS DH. DH without authentication. It is vulnerable to MIM.



Certificate & server_key_exchange

- Certificate: always requested but anonymous Diffie-Hellman
- Server_key_exchange
  - Not requested in Fixed Diffie-Hellman and RSA
  - The format depends on the chosen key exchange algorithm
  - Requested in
    - Anonymous Diffie-Hellman → (p, g, Y_{svr})
    - Ephemeral Diffie-Hellman → $\langle p, g, Y_{svr}\rangle_{svr}$
    - RSA-based where the server has RSA-signing-key → $\langle TempPubK_{svr}\rangle_{svr}$

The server_key_echange message is not requested in Fixed Diffie-Hellman and RSA. In these cases, certificates are transmitted in the certificate message. In Anonymous Diffie-Helmman, the server_key_echange message carries (p, g, Y_{svr}). In Ephemeral Diffie-Hellman, the server_key_echange message carries $\langle p, g, Y_{svr}\rangle_{svr}$ digitally signed by the server. The certificate for verifying the digital signature was conveyed by the certificate message. In RSA-based key exchange where the server has RSA-signing key, the server_key_echange message conveys an ephemeral temporary public key $TempPubK_{svr}$ which features and Ephemeral RSA (RSAE). More precisely, the message carries $\langle TempPubK_{svr}\rangle_{svr}$, namely the public key digitally signed by the server. The certificate for verifying the digital signature was conveyed by the certificate message.

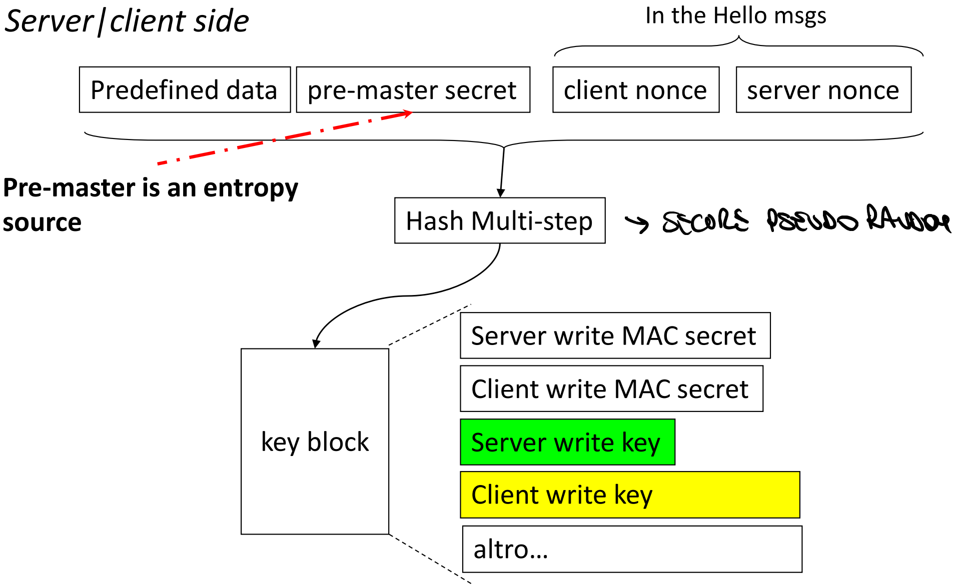


Client_key_exchange message

- 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 (client → server)

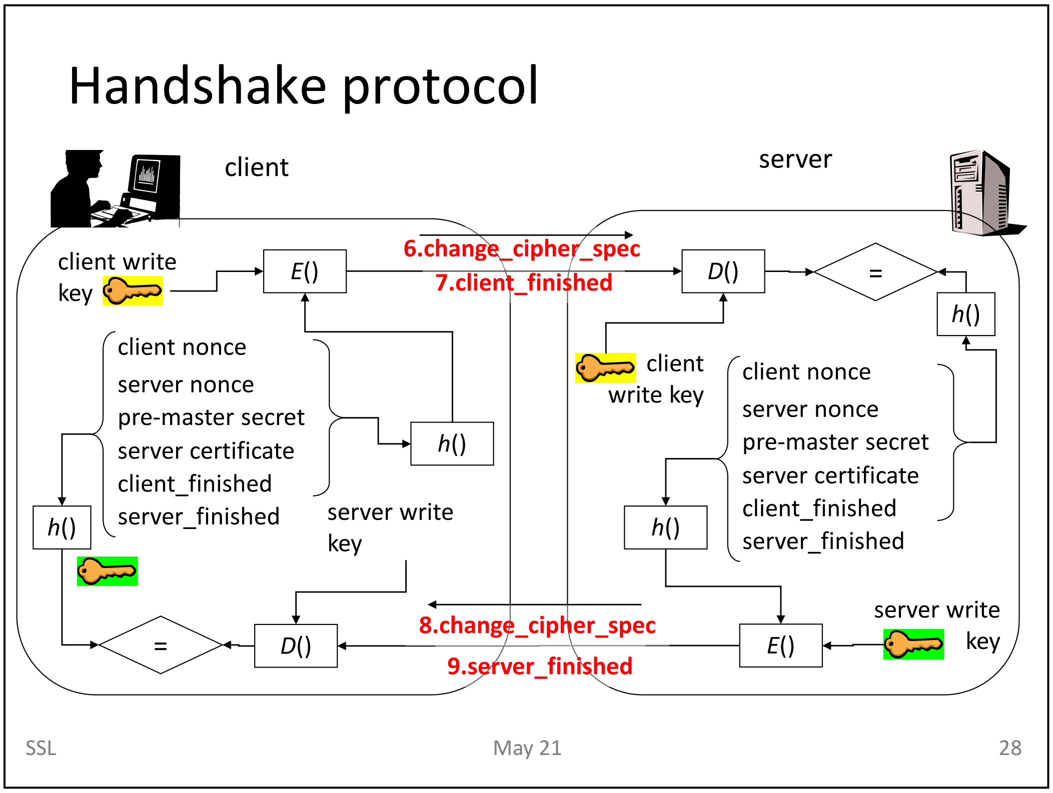
Key generation

Server|client side



Pre-master is an entropy source

→ SECURE PSEUDO RANDOM BYTE GENERATOR



Handshake protocol

client

server

client write key

E()

6.change_cipher_spec

7.client_finished

D()

=

h()

client write key

client nonce
server nonce
pre-master secret
server certificate
client_finished
server_finished

h()

h()

client nonce
server nonce
pre-master secret
server certificate
client_finished
server_finished

h()

server write key

=

D()

8.change_cipher_spec

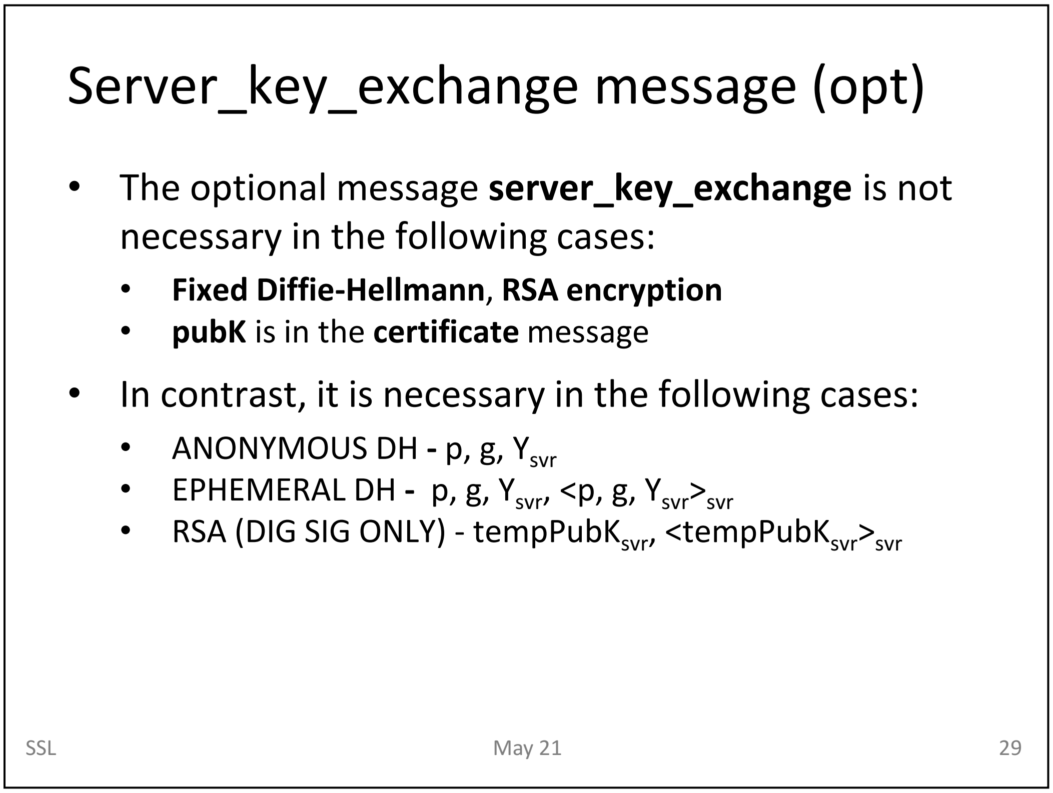
9.server_finished

E()

server write key

SSL May 21 28

The hash is computed on the messages received until that moment.
The hash is actually computed twice, one with MD5 and one with SHA-1. Both digests are then encrypted.



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}, $<p, g, Y_{svr}>_{svr}$
  - RSA (DIG SIG ONLY) - $tempPubK_{svr}$, $<tempPubK_{svr}>_{svr}$

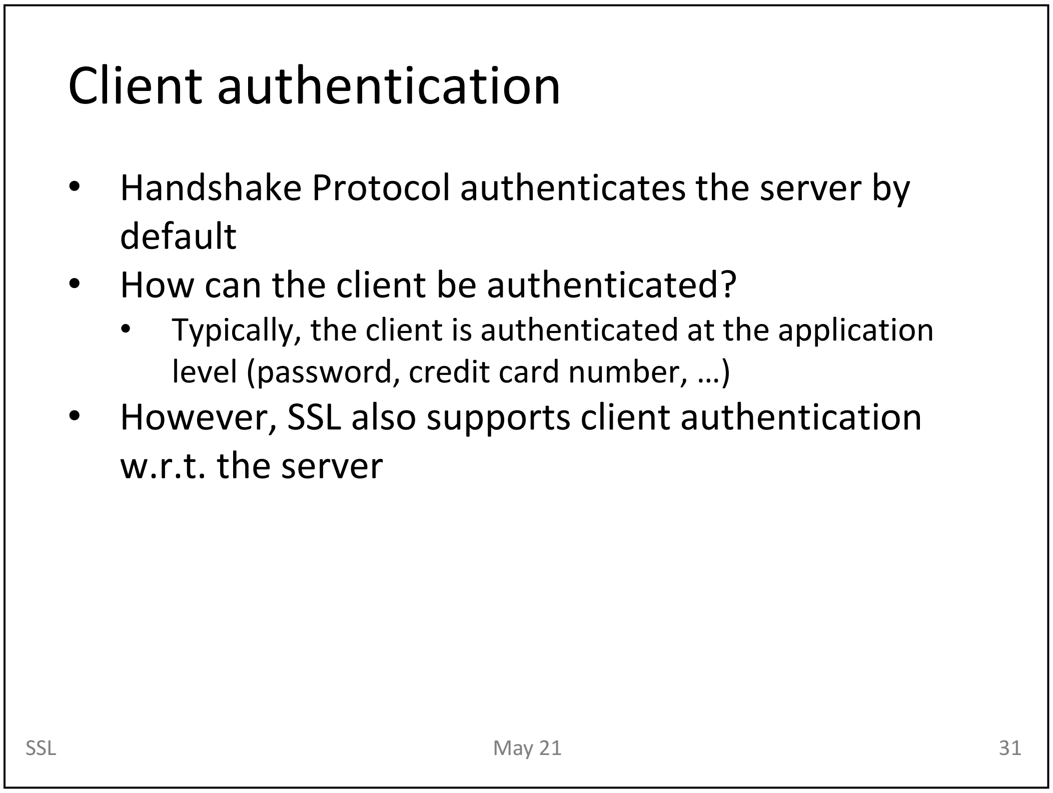
SSL May 21 29

The server_key_exchange message hasn’t been shown in the example. Its presence depends on the key establishment method selected by means of the cipher suite. If the method is fixed Diffie-Hellmann or RSA encryption, than the server_key_exchange mesdsage is not necessary because the server’s public key is carried by the certificate message. In fact, the server_key_exchange message is necessary in the following cases:

- ANONYMOUS DH: The certificate message is absent; the server_key_message message conveys $(p, g, Y)_{svr}$. These parameters, in particular Y are not authenticated.
- EPHEMERAL DH: The server_key_exchange message conveys $<p, g, Y>_{svr}$ which denotes that parameters p, g and Y are digitally signed by the server.
- RSA DIGITAL SIGNATURE: the server generates a temporarily pair $(privK, pubK)_{temp}$ and uses the server_key_exchange message to convey $pubK_{temp}$. Actually, the message conveys $<pubk_{temp}>_{svr}$ which denotes that the contents are digitally signed by the server. The digital signature involves also ClientHello.random and ServerHello.random in order to avoid replay attacks. Furthermore, it actually digitally signs the hash computed by means of SHA-1 (MD5) in the case of DSS (RSA) digital signature.

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

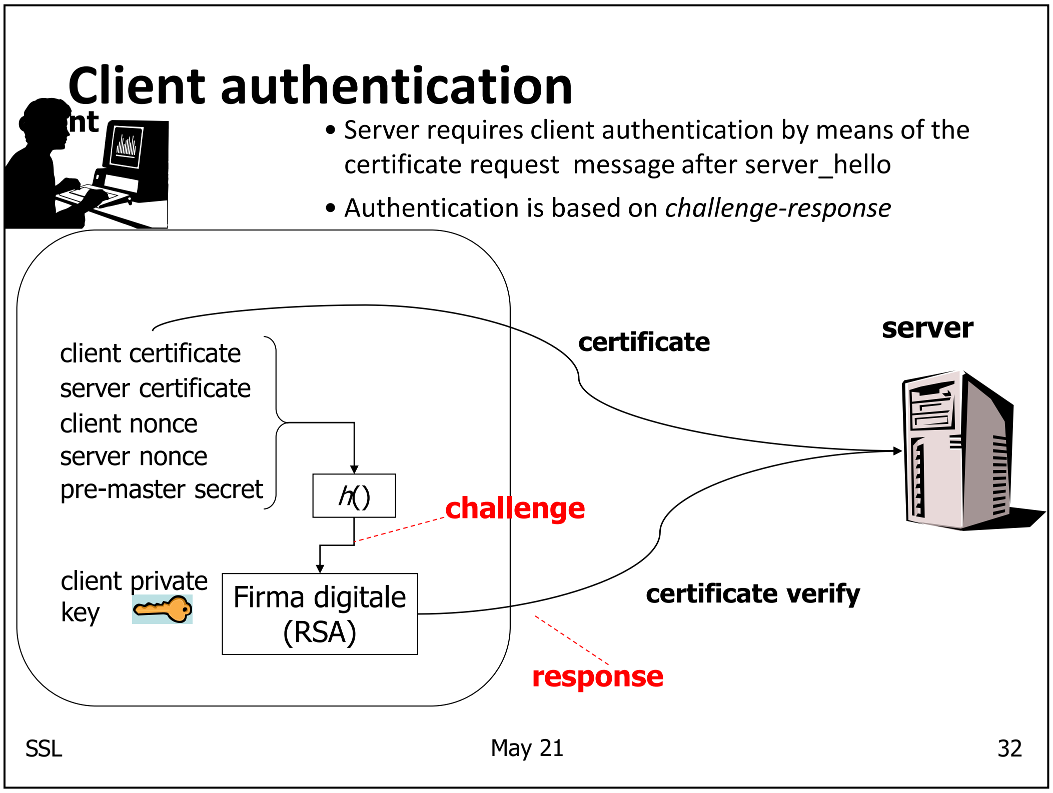
- Handshake Protocol authenticates the server by default
- How can the client be authenticated?
  - Typically, the client is authenticated at the application level (password, credit card number, …)
- However, SSL also supports client authentication w.r.t. the server

SSL May 21 31

Quanto è sicuro il canale SSL?

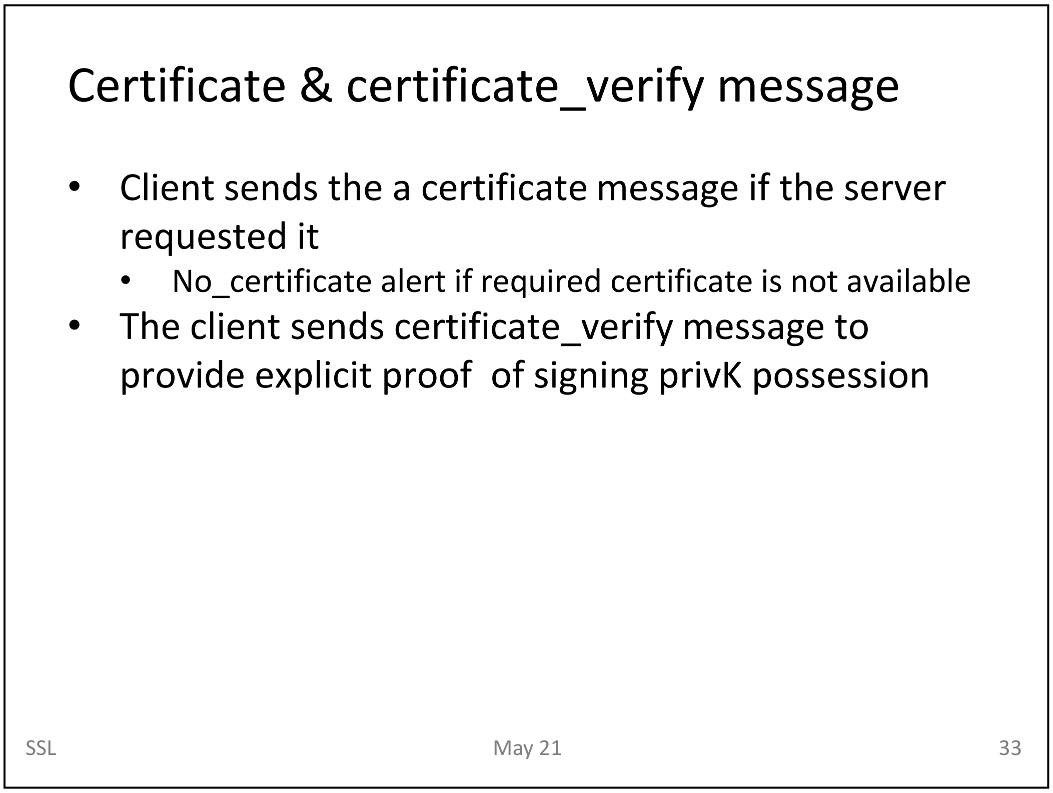
A causa delle restrizioni imposte dalla precedente normativa U.S. sulla esportazione di materiale crittografico, le vecchie versioni dei browser supportavano solo chiavi di sessione a 40 bit (e chiavi pubbliche a 512 bit) invece dei 128 bit supportati dai browser più recenti

I vecchi browser sono ancora in uso; molti utenti sono ignari del problema



Client authentication

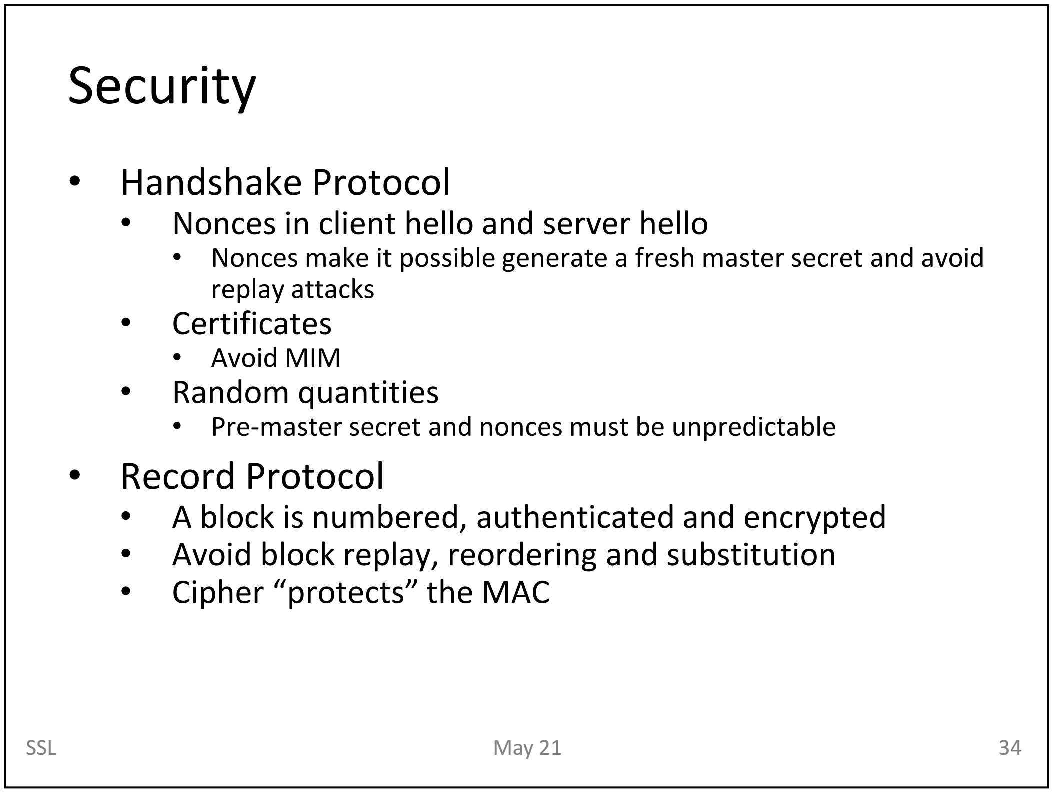
- Server requires client authentication by means of the certificate request message after server_hello
- Authentication is based on *challenge-response*



Certificate & certificate_verify message

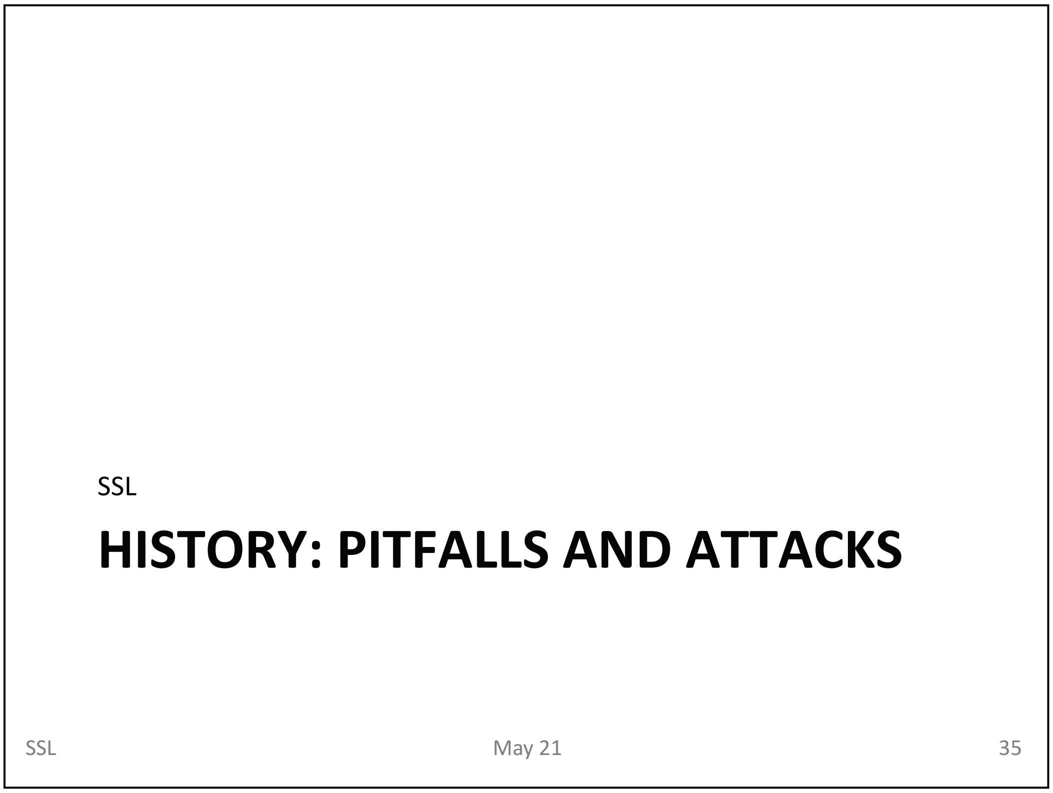
- Client sends the a certificate message if the server requested it
  - No_certificate alert if required certificate is not available
- The client sends certificate_verify message to provide explicit proof of signing privK possession

SSL May 21 33



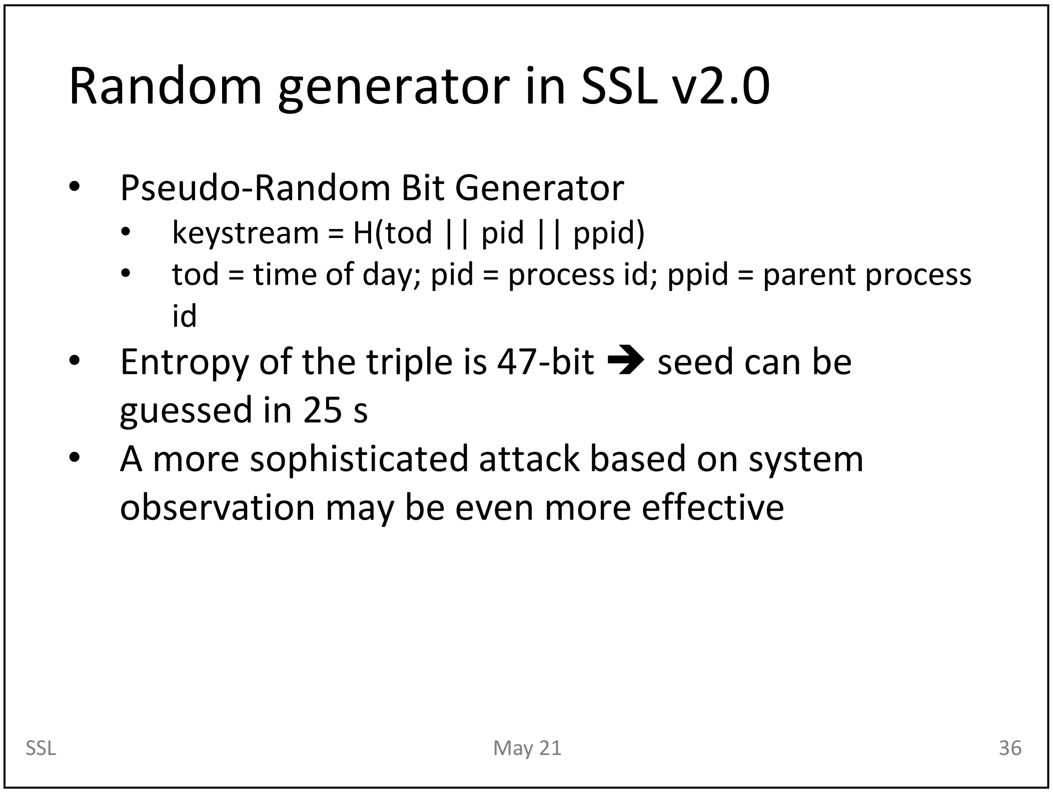
Security

- 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 unpredictable
- Record Protocol
  - A block is numbered, authenticated and encrypted
  - Avoid block replay, reordering and substitution
  - Cipher “protects” the MAC



SSL

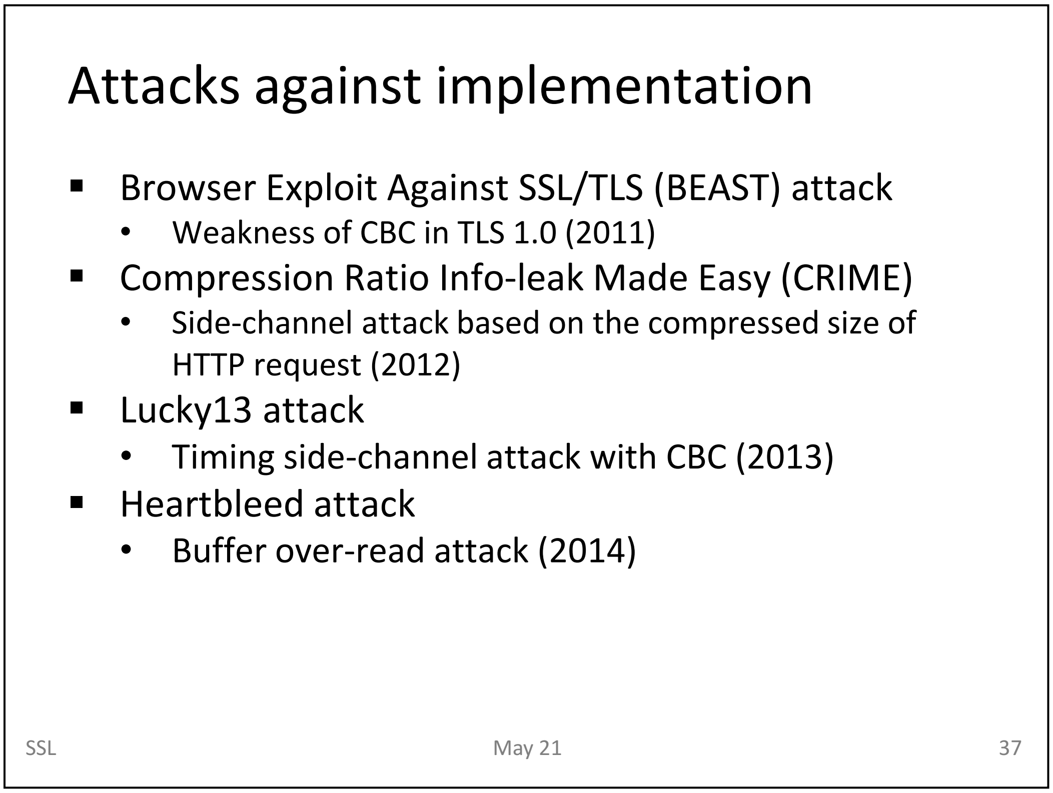
HISTORY: PITFALLS AND ATTACKS



Random generator in SSL v2.0

- 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

A pseudo-random bit generator (PRBG) takes a seed as input and generates a keystream, *keystream* = PRBG(*seed*). If the seed is predictable than the bit stream is predictable. Entropy gives a measure of predictability of a given information.
Entropy is the measure of how many bits of a given information are unpredictable. An information represented on *L* bits may have an entropy which ranges from 0 (totally predictable) to 2^L (completely unpredictable). If the entropy of a given information is *x* then you need $O(2^x)$ trials to guess it.
Nowadays, the seed of a secure PRBG (SPRBG) must have an entropy not smaller than 64-bits.
Typically, a library can tell you if 1) a generator has been seeded, and 2) a seed is on a large enough number of bits. However, a library cannot tell you if the seed is good, i.e., it’s entropy is high enough.



Attacks against implementation

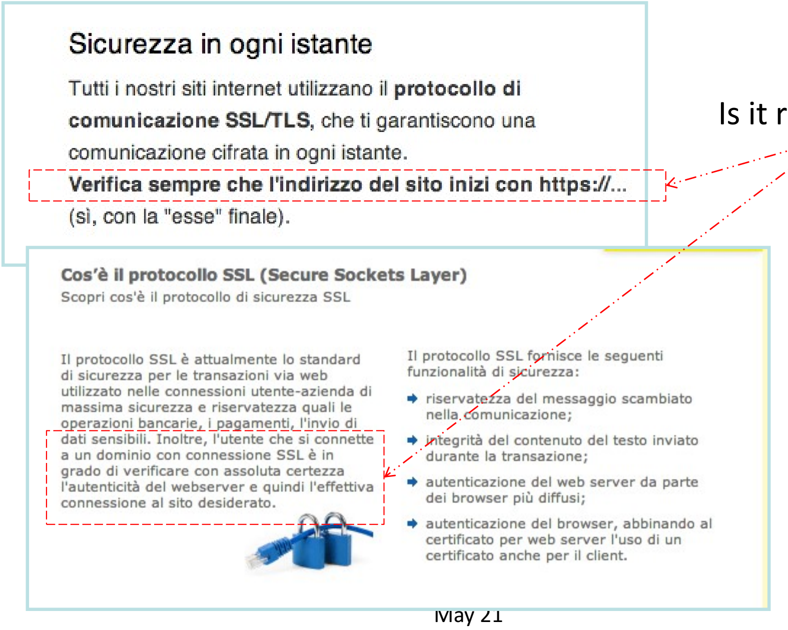
- Browser Exploit Against SSL/TLS (BEAST) attack
  - Weakness of CBC in TLS 1.0 (2011)
- Compression Ratio Info-leak Made Easy (CRIME)
  - Side-channel attack based on the compressed size of HTTP request (2012)
- Lucky13 attack
  - Timing side-channel attack with CBC (2013)
- Heartbleed attack
  - Buffer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

SSL May 21 37

SSL

ON USING SSL IN E-COMMERCE

SSL in action



Sicurezza in ogni istante

Tutti i nostri siti internet utilizzano il **protocollo di comunicazione SSL/TLS**, che ti garantiscono una comunicazione cifrata in ogni istante.
Verifica sempre che l'indirizzo del sito inizi con https://...
(sì, con la "esse" finale).

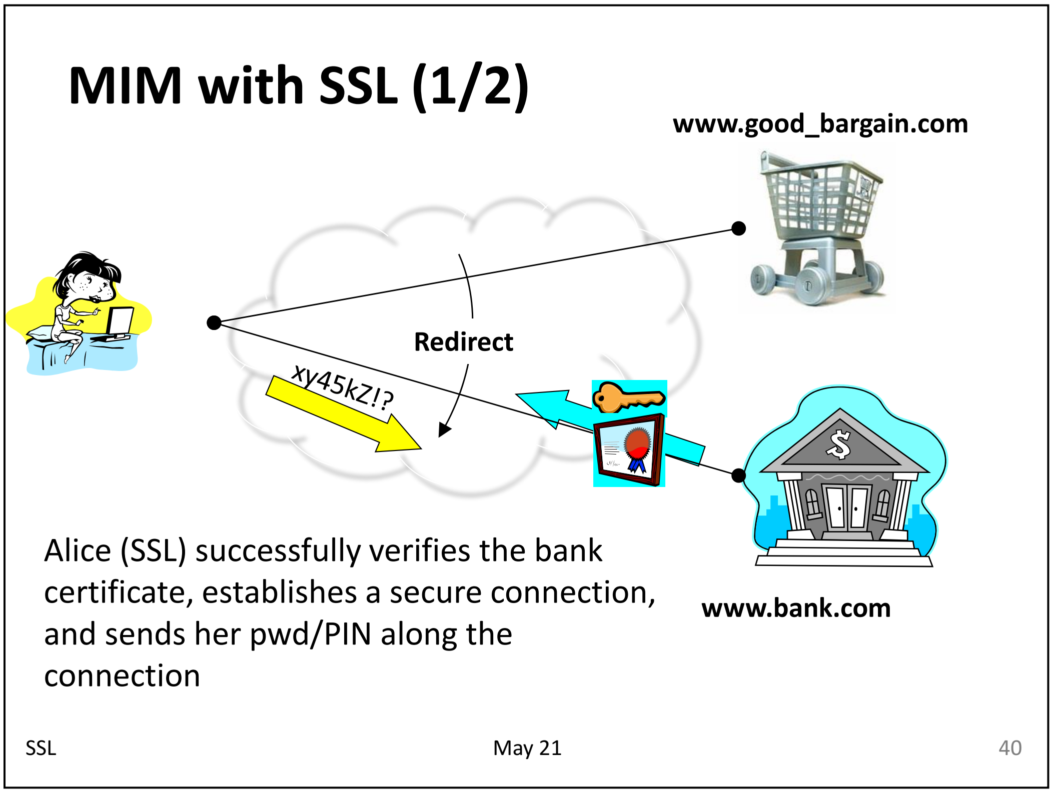
Cos'è il protocollo SSL (Secure Sockets Layer)
Scopri cos'è il protocollo di sicurezza SSL

Il protocollo SSL è attualmente lo standard di sicurezza per le transazioni via web utilizzato nelle connessioni utente-azienda di massima sicurezza e riservatezza quali le operazioni bancarie, i pagamenti, l'invio di dati sensibili. Inoltre, l'utente che si connette a un dominio con connessione SSL è in grado di verificare con assoluta certezza l'autenticità del webserver e quindi l'effettiva connessione al sito desiderato.

Il protocollo SSL fornisce le seguenti funzionalità di sicurezza:

- riservatezza del messaggio scambiato nella comunicazione;
- integrità del contenuto del testo inviato durante la transazione;
- autenticazione del web server da parte dei browser più diffusi;
- autenticazione del browser, abbinando al certificato per web server l'uso di un certificato anche per il client.

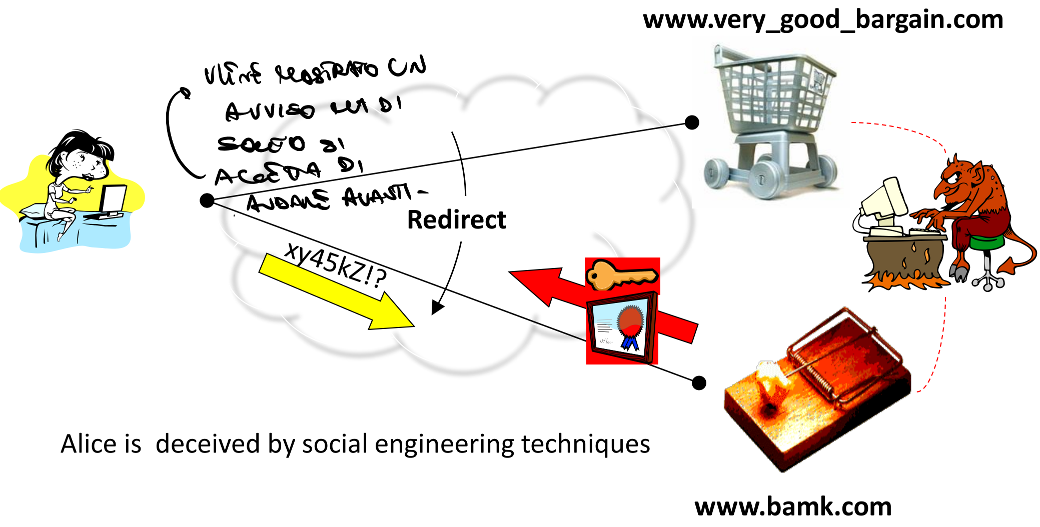
Is it really true?



MIM with SSL (1/2)

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

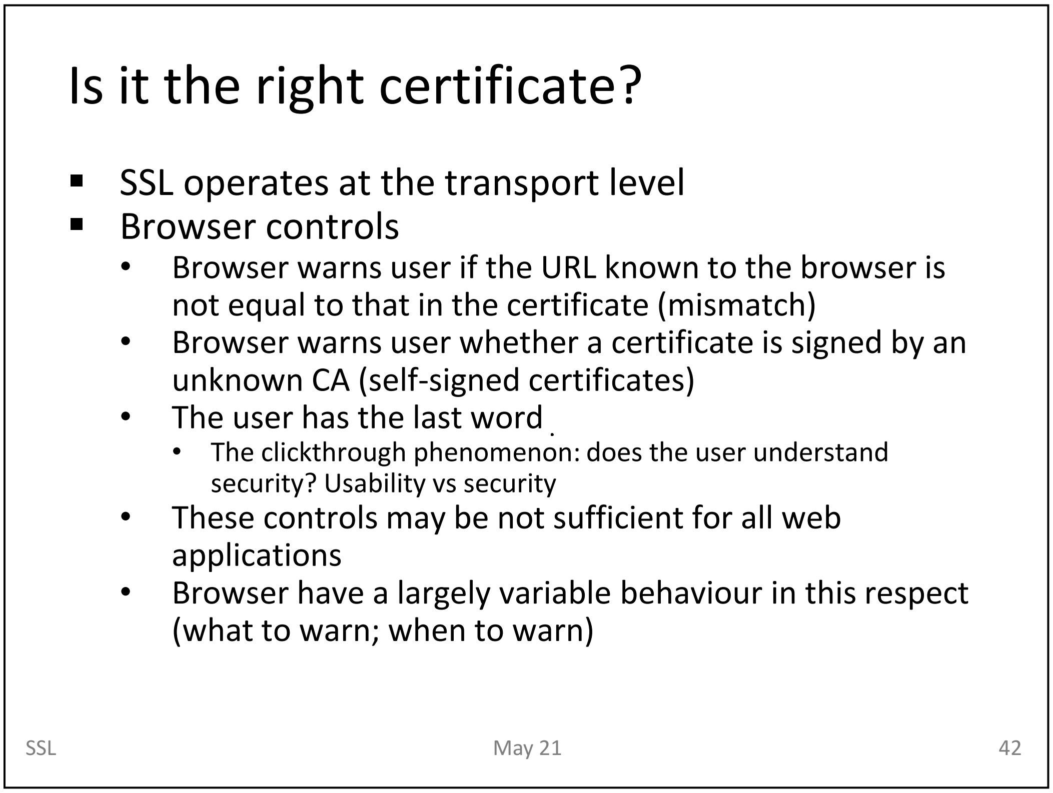
MIM with SSL (2/2)



VIENE MOSTRATO UN AVVISO LUI DI SOLITO DI ACCETTA DI ANDARE AVANTI -

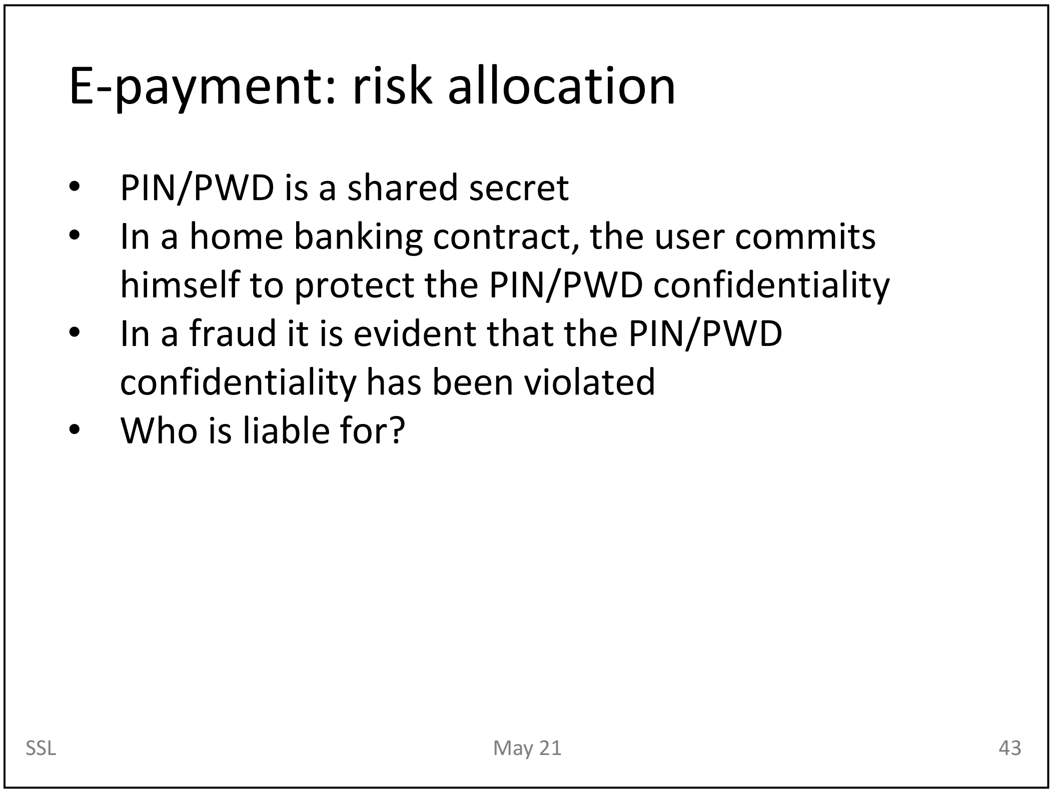
Alice is deceived by social engineering techniques

Social engineering, in the context of information security, is understood to mean **the art of manipulating people into performing actions or divulging confidential information**.



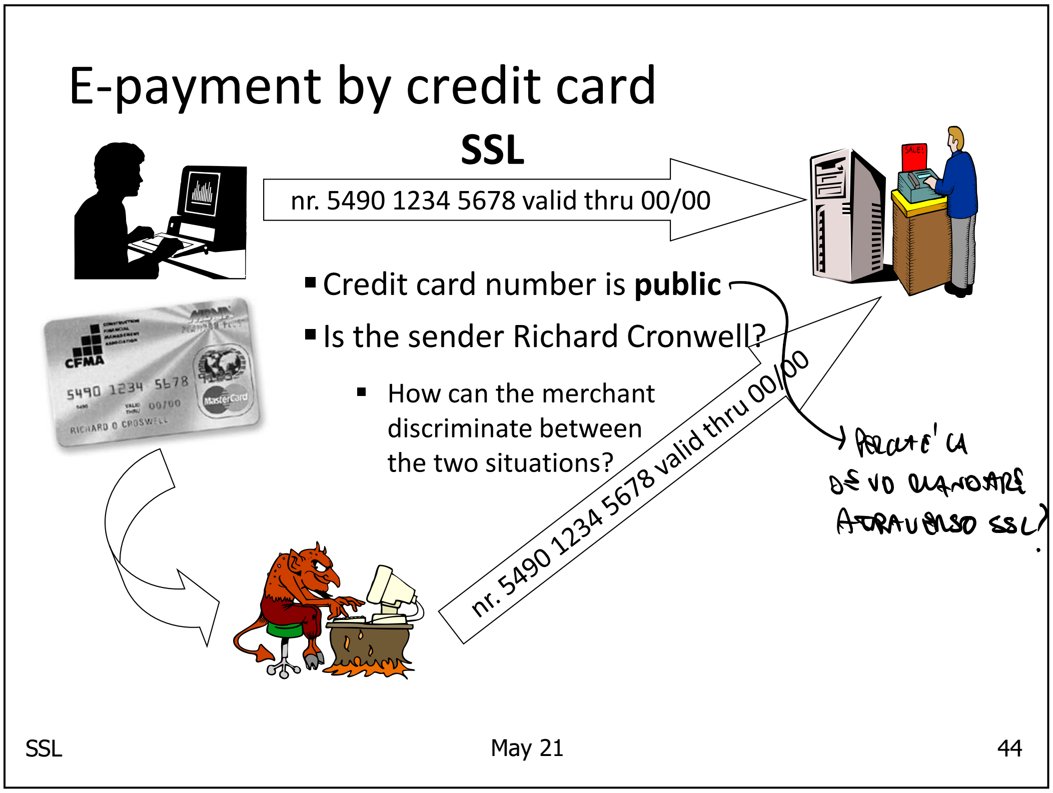
Is it the right certificate?

- SSL operates at the transport 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)



E-payment: 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?

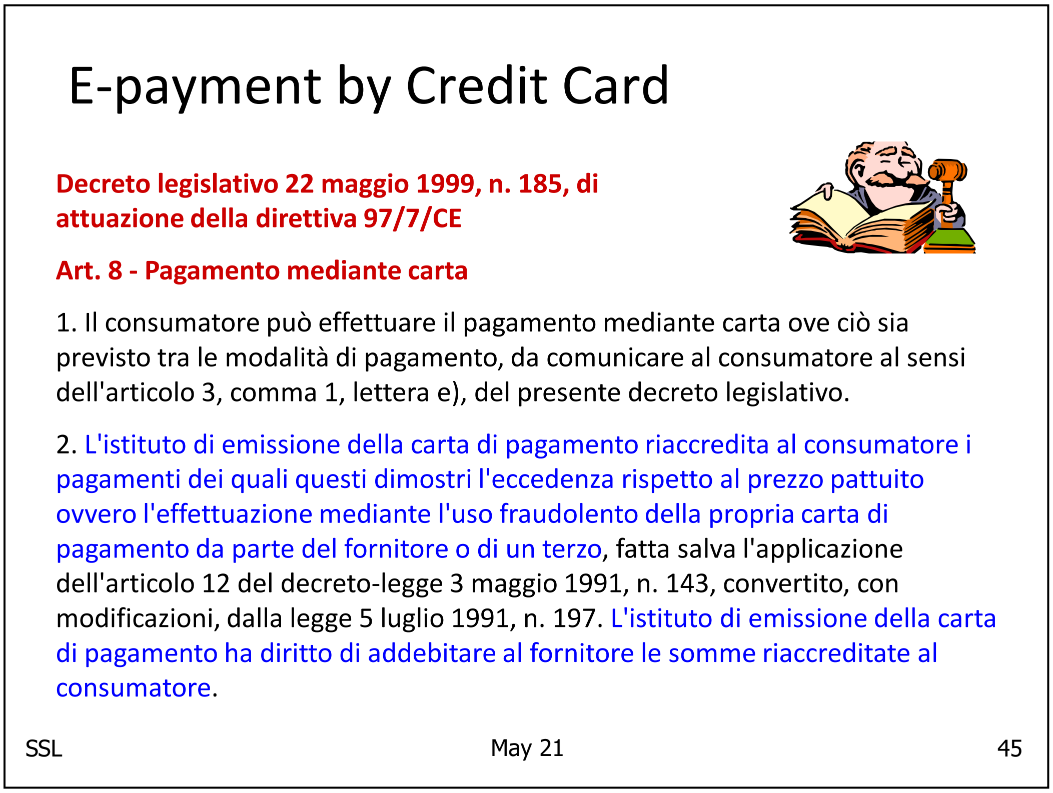


E-payment by credit card

SSL
nr. 5490 1234 5678 valid thru 00/00
nr. 5490 1234 5678 valid thru 00/00
5490 1234 5678
00/00
RICHARD O CROSWELL
CFMA
MasterCard

- Credit card number is **public**
- Is the sender Richard Cronwell?
  - How can the merchant discriminate between the two situations?

PERCHE' LA DEVO MANDARE ATTRAVERSO SSL?



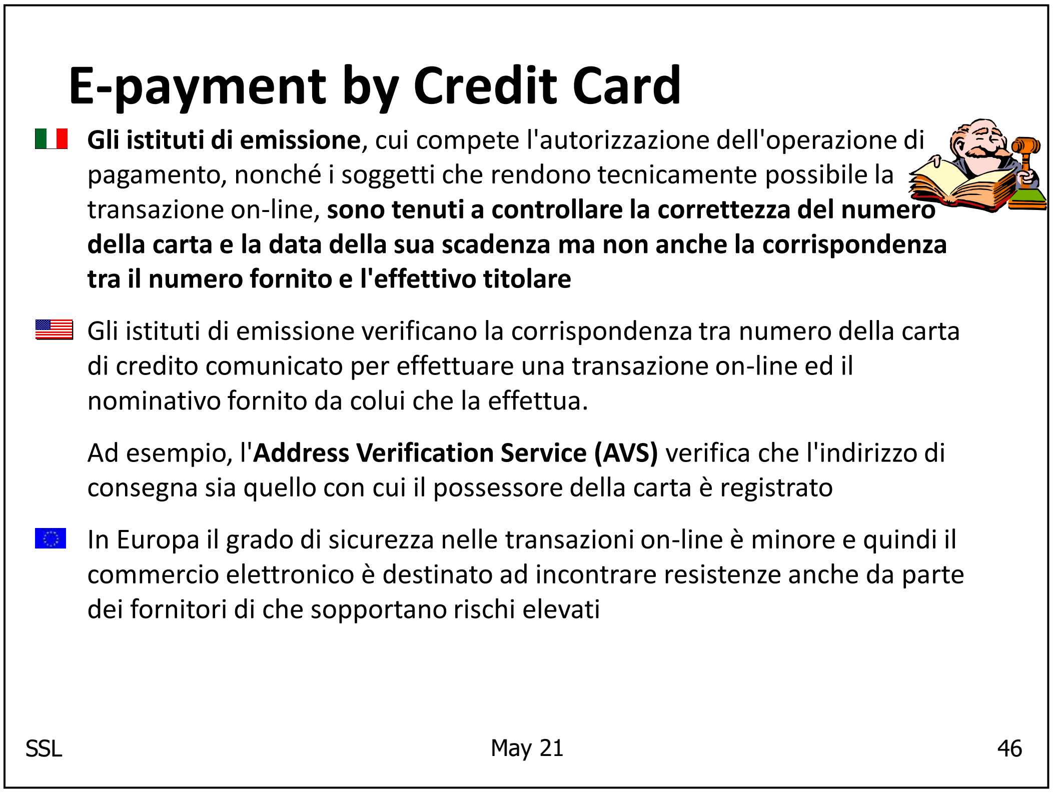
E-payment by Credit Card

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.

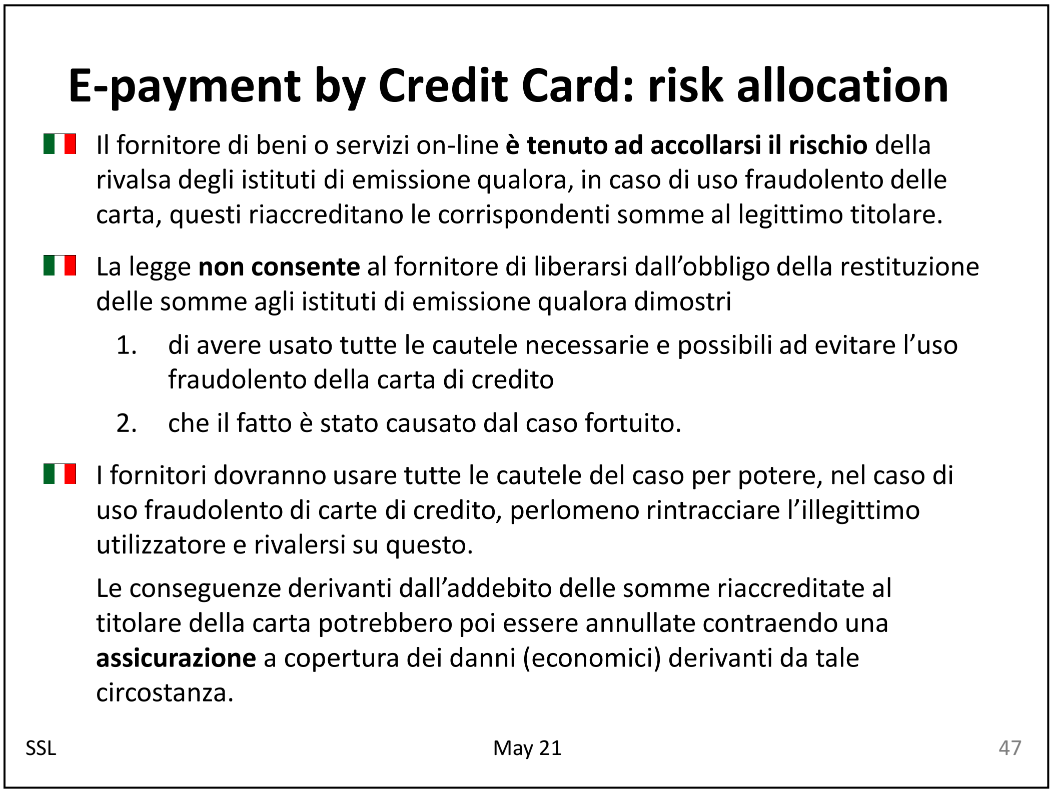
2. L'istituto di emissione della carta di pagamento riaccredita al consumatore i pagamenti dei quali questi dimostri l'eccedenza rispetto al prezzo pattuito ovvero l'effettuazione mediante l'uso fraudolento della propria carta di pagamento da parte del fornitore o di un terzo, fatta salva l'applicazione 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



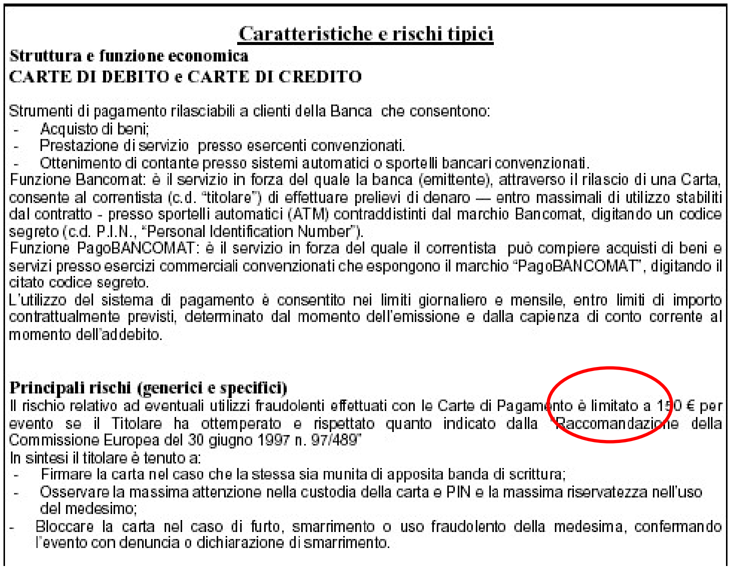
E-payment by Credit Card: risk allocation

- 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.

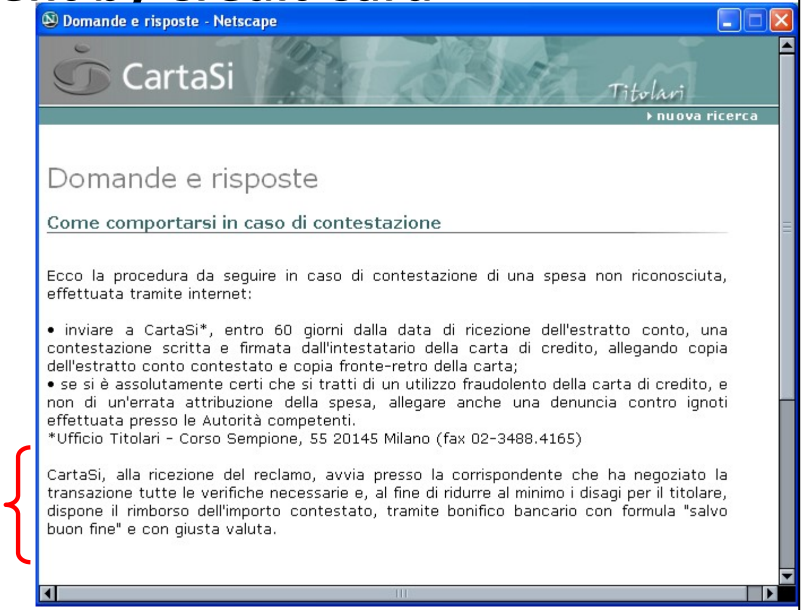
Principali rischi (generici e specifici)

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



Domande e risposte

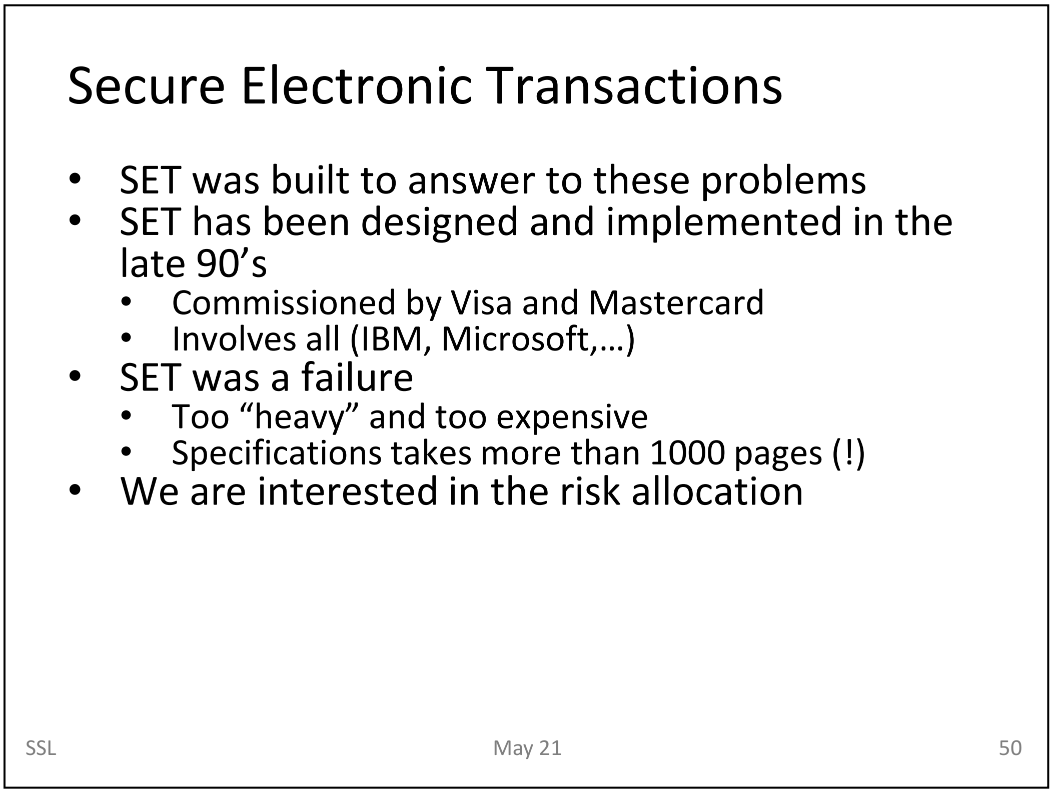
Come comportarsi in caso di contestazione

Ecco la procedura da seguire in caso di contestazione di una spesa non riconosciuta, effettuata tramite internet:

- inviare a CartaSi*, entro 60 giorni dalla data di ricezione dell'estratto conto, una contestazione scritta e firmata dall'intestatario della carta di credito, allegando copia dell'estratto conto contestato e copia fronte-retro della carta;
- se si è assolutamente certi che si tratti di un utilizzo fraudolento della carta di credito, e non di un'errata attribuzione della spesa, allegare anche una denuncia contro ignoti effettuata presso le Autorità competenti.

*Ufficio Titolari - Corso Sempione, 55 20145 Milano (fax 02-3488.4165)

CartaSi, alla ricezione del reclamo, avvia presso la corrispondente che ha negoziato la transazione tutte le verifiche necessarie e, al fine di ridurre al minimo i disagi per il titolare, dispone il rimborso dell'importo contestato, tramite bonifico bancario con formula "salvo buon fine" e con giusta valuta.

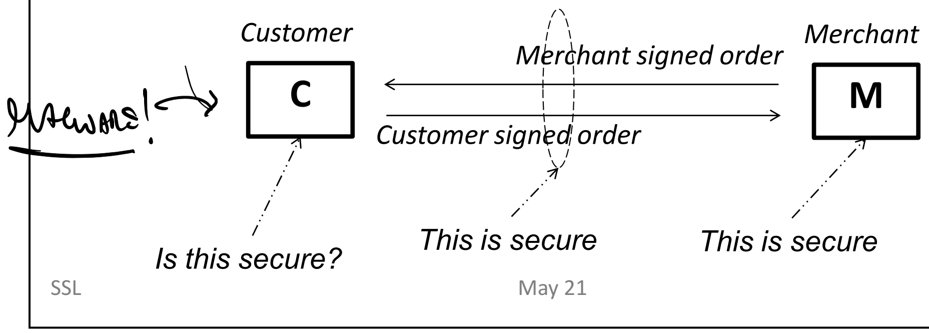


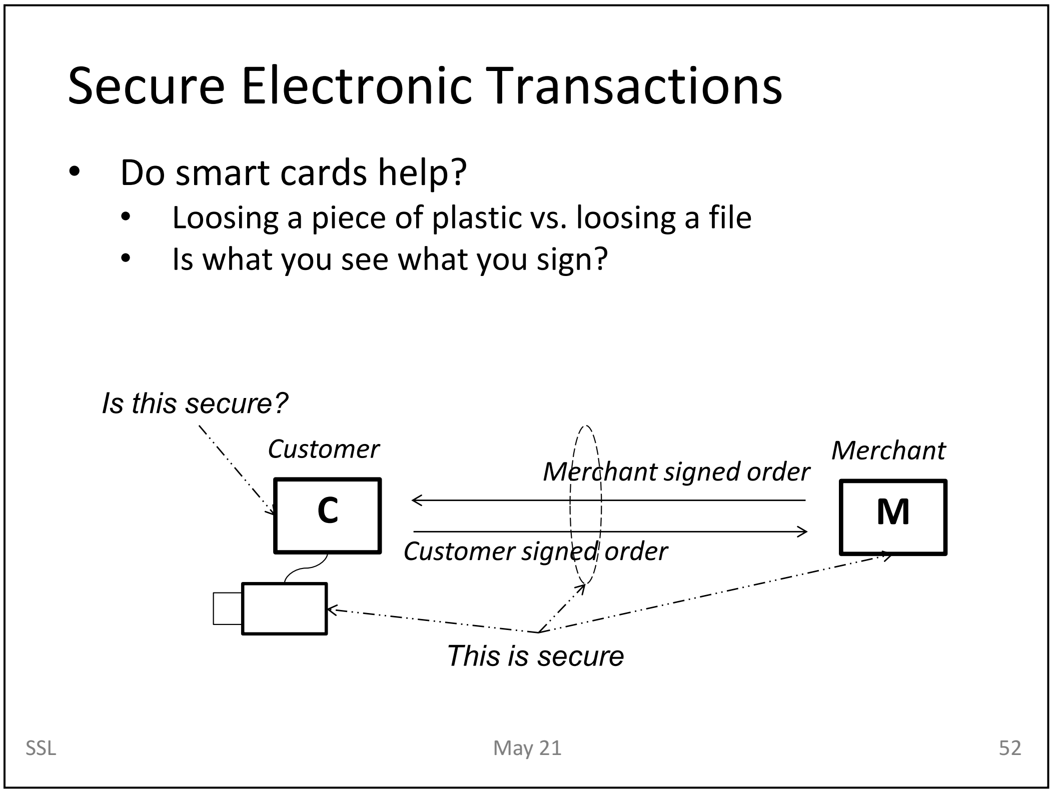
Secure Electronic Transactions

- 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” and too expensive
  - Specifications takes more than 1000 pages (!)
- We are interested in the risk allocation

Secure Electronic Transactions

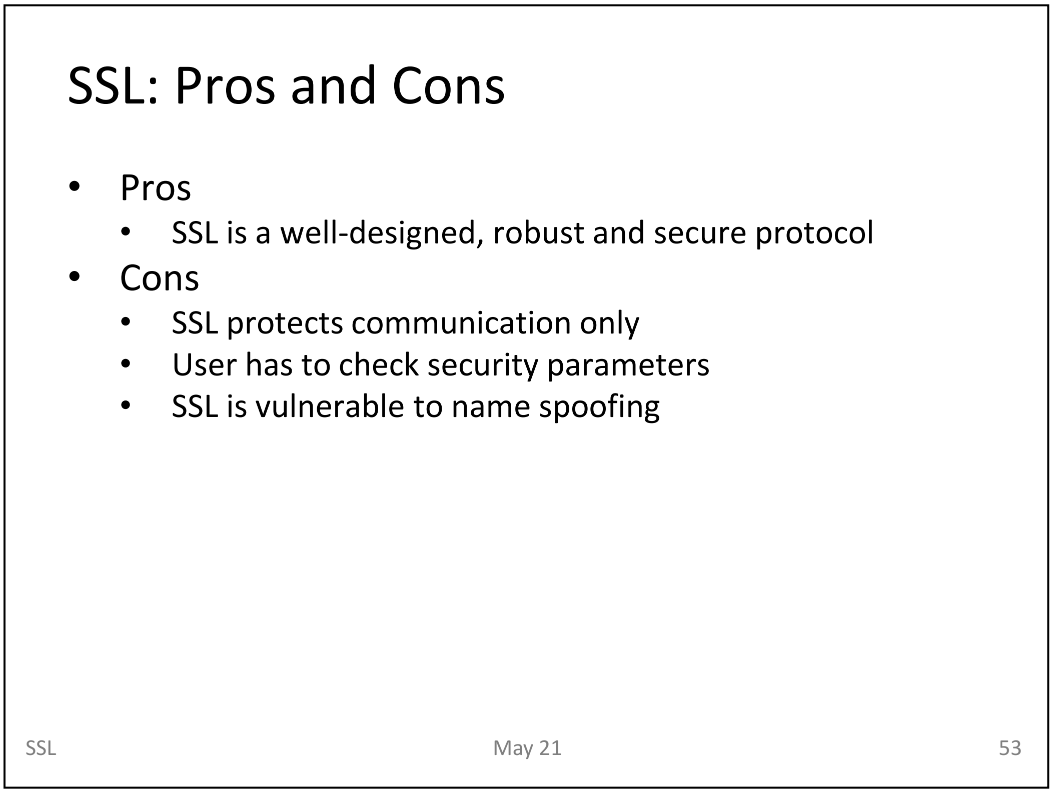
- SET requires a PKI in place
- A (privK, pubK) pair is stored at M and C
- *If an order is signed by your key you cannot repudiate it*
  - **The risk is allocated on the customer**
- M and C are assumed trusted devices!
  - Stealing a privK is equivalent to stealing a file





Secure Electronic Transactions

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



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