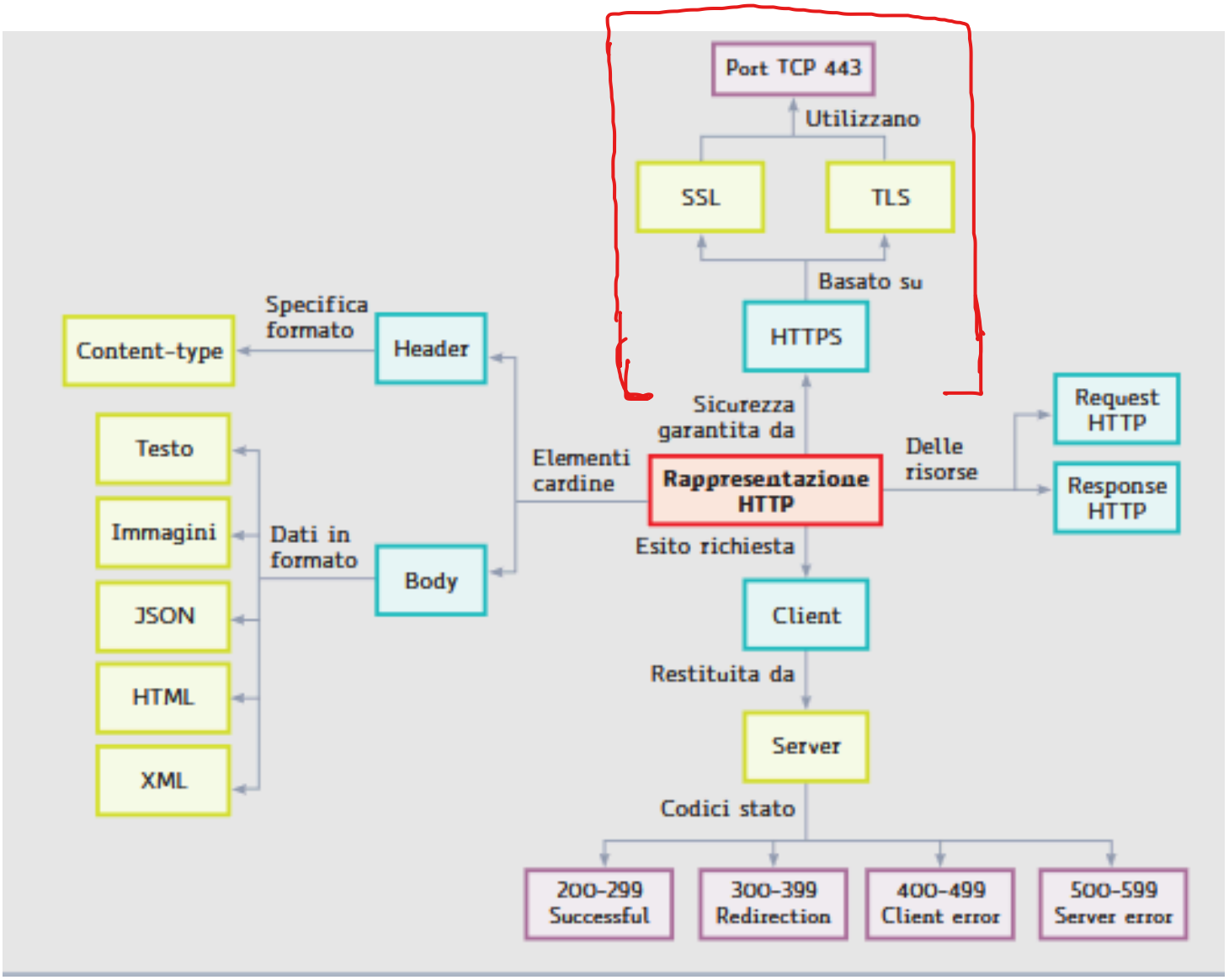
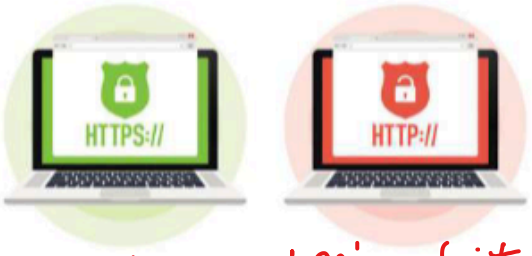

Port TCP 443
Utilizzano
SSL
TLS
Basato su
HTTPS
Sicurezza garantita da
Rappresentazione HTTP
Elementi cardine
Delle risorse
Request HTTP
Response HTTP
Esito richiesta
Client
Restituita da
Server
Codici stato
200-299 Successful
300-399 Redirection
400-499 Client error
500-599 Server error
Specifica formato
Header
Content-type
Dati in formato
Body
Testo
Immagini
JSON
HTML
XML

La sicurezza con HTTPS

Il protocollo **HTTPS** (Hyper Text Transfer Protocol over Ssl) è una variante successiva del protocollo HTTP che rende più sicure le connessioni client server mediante tecniche di cifratura dei dati.

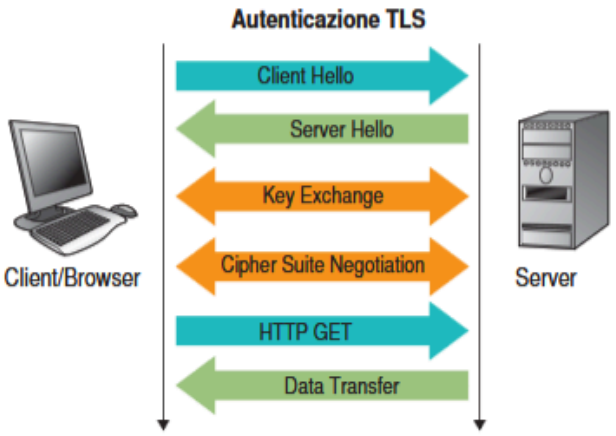


HTTPS
Mediante il protocollo HTTPS i dati scambiati tra client e server sono protetti da tentativi di eavesdropping, cioè di ascolto senza consenso, e di tampering, che è la manomissione della comunicazione falsandone i contenuti. Con il tampering gli utenti non si accorgono di essere in comunicazione con altri server che possono carpirne i dati.

L'HTTPS utilizza, oltre al protocollo TCP/IP, i protocolli SSL (Secure Sockets Layer) e TLS (Transport Layer Security) che si occupano della crittografia e dell'autenticazione dei dati trasmessi. La comunicazione con SSL/TLS è basata su un algoritmo di cifratura a chiave simmetrica (es. AES), la cui chiave è generata da una delle due parti con un algoritmo a chiave asimmetrica (es. RSA). Questo tipo di comunicazione garantisce che solamente il client e il server siano in grado di conoscere il contenuto della comunicazione e impedisce a terze parti di leggere, inserire o modificare i messaggi scambiati tra i due interlocutori.

> Il protocollo HTTPS utilizza di default la porta TCP 443.

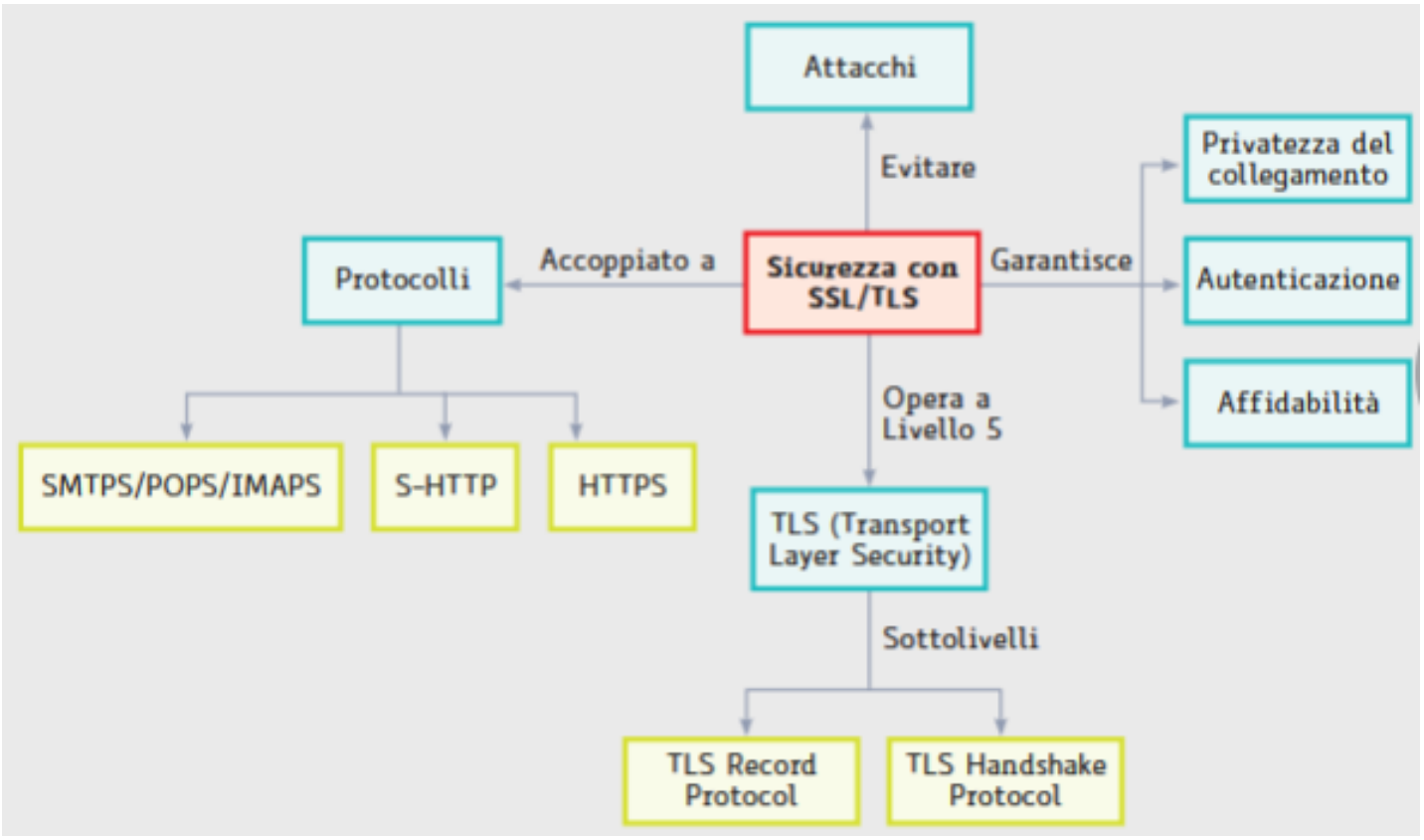
Nell'utilizzo tipico di un browser, l'autenticazione TLS avviene in modo unilaterale: il client conosce l'identità del server e soltanto il server viene autenticato: il client rimane anonimo e quindi non autenticato. L'autenticazione del server è molto utile per il software di navigazione e per l'utente, il browser valida il certificato del server controllando la firma digitale tra i certificate authority conosciuti, applicando una cifratura a chiave pubblica.



Al termine dell'autenticazione il client (browser) indica una connessione sicura mostrando solitamente un'icona a forma di lucchetto. Tuttavia questa autenticazione non è sufficiente per garantire che il sito al quale è collegato il client sia effettivamente quello richiesto: per assicurarsi è necessario analizzare il contenuto del certificato rilasciato e controllarne la catena di certificazione.

Solo le Certificate Authority (CA) possono generare certificati validi con un URL incorporato in modo che venga confrontata con l'URL richiesto.

Il protocollo TLS permette anche un'autenticazione bilaterale, dove entrambe le parti (server e client browser) si autenticano in modo sicuro scambiandosi i relativi certificati. Questa autenticazione, chiamata MA (Mutual Authentication), richiede che anche il client possieda un proprio certificato digitale.



Il protocollo SSL/TLS

Il protocollo SSL garantisce la sicurezza del collegamento mediante tre funzionalità fondamentali:

- privatezza del collegamento: la riservatezza del collegamento viene garantita mediante algoritmi di crittografia a chiave simmetrica (per esempio DES e RC4);
- autenticazione: l'autenticazione dell'identità viene effettuata con la crittografia a chiave pubblica (per esempio RSA e DSS): in questo modo si garantisce ai client di comunicare con il server corretto, introducendo a tale scopo anche meccanismi di certificazione sia del server sia del client;
- affidabilità: il livello di trasporto include un controllo sull'integrità del messaggio con un sistema detto MAC (Message Authentication Code) che utilizza funzioni hash sicure come SHA e MD5: avviene la verifica di integrità sui dati spediti in modo da avere la certezza che non siano stati alterati durante la trasmissione.

> ⚠ Tutti i dati trasmessi dal client al server, e viceversa, vengono cifrati.

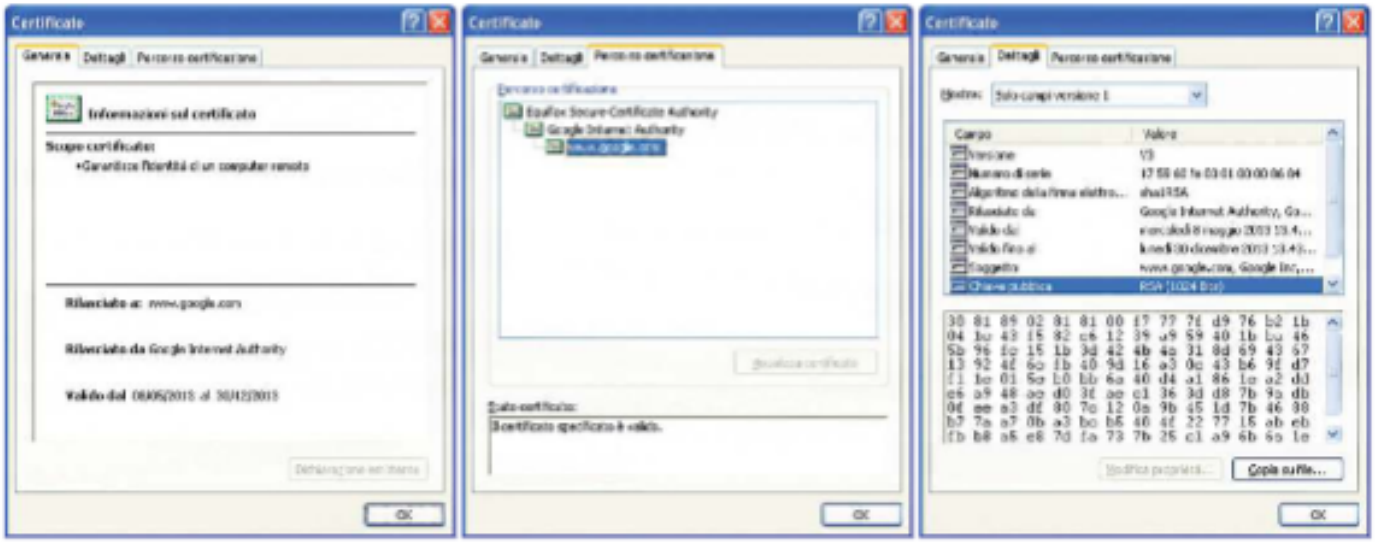
Viene accoppiato molto spesso ad altri protocolli che lavorano a livello applicativo ottenendo meccanismi sicuri come:

- HTTPS (RFC 2818): si ottiene combinando HTTP con SSL/TLS ed è usato per proteggere i dati sensibili inviati da e per i server web: usa una porta 443 e https:// come prefisso per gli URL;
- S-HTTP (RFC 2660): è poco diffuso e incapsula i dati HTTP in un messaggio crittografato secondo un formato MIME apposito o il formato CMS (Cryptographic Message Syntax);
- SMTPS/POPS/IMAPS: utilizzati per proteggere il contenuto delle email inviate (lavorano sulle porte 465/TCP per SMTPS, 995/TCP per POP3S e 993/TCP per IMAPS).

HTTPS

Gli indirizzi dei siti protetti con SSL iniziano per https:// e hanno alla loro sinistra l'immagine di un lucchetto: facendo clic su di esso viene visualizzata una finestra contenente le autorizzazioni e le caratteristiche della connessione.

Nella prima parte che riguarda l'identità è possibile visualizzare nel dettaglio le informazioni sul certificato (standard X.509 v3) che riportiamo nelle immagini seguenti.



Ogni certificato deve contenere almeno:
- il nome/indirizzo del server, affinché il client possa confrontarlo con il nome della macchina a cui è collegato;
- la chiave pubblica del server;
- il nome dell'autorità certificante.

Il certificato è firmato digitalmente dall'autorità certificante che quindi si fa garante dell'autenticità delle informazioni contenute nel certificato. Se l'amministratore di un sito web decide di utilizzare TLS ha necessariamente bisogno di essere certificato da un CA, generalmente da una root authority, che lo rilascia a pagamento stipulando generalmente contratti annuali (per esempio VeriSign stipula contratti a partire da $399/anno). È anche possibile generare "in proprio" un certificato, detto self-signed (chi deve essere certificato funge anche da autorità certificante di se stesso), utilizzando per esempio librerie openssl: molti server lo utilizzano per assicurare la riservatezza delle connessioni mediante la cifratura non potendo però garantire l'autenticazione del "proprietario"!

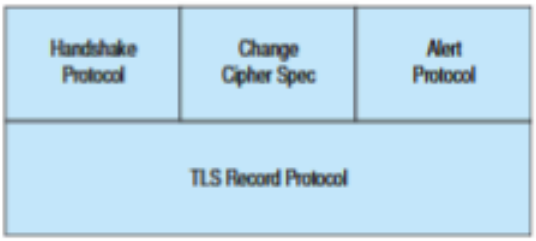
Il funzionamento di TLS

TSL è un protocollo di livello 5 (sessione) che opera quindi al di sopra del livello di trasporto composto da due livelli: ISO/OSI

- TLS Record Protocol: opera a livello più basso, direttamente al di sopra di un protocollo di trasporto affidabile come il TCP ed è utilizzato per i protocolli del livello superiore, tra cui l'Handshake Protocol, offrendo in questo modo i servizi di sicurezza;
- TLS Handshake Protocol: si occupa della fase di negoziazione in cui si autentica l'interlocutore e si stabiliscono le chiavi segrete condivise, organizzato in tre sottoprotocolli:
  - Handshake Protocol;
  - Change Cipher Spec Protocol;
  - Alert Protocol.

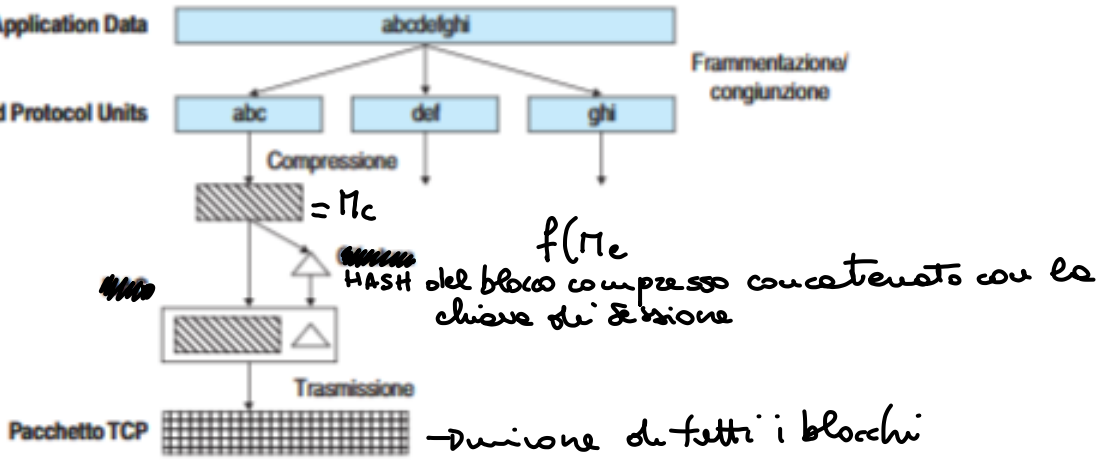
In sintesi, TLS definisce il Record Protocol per trasferire i dati dell'applicazione e stabilisce la sessione utilizzando l'Handshake Protocol.

L'architettura completa di TLS è mostrata qui a lato.



TLS Record Protocol

Il TLS Record Protocol prende i dati dal livello superiore, li suddivide in blocchi, eventualmente li comprime, calcola ~~il MAC~~ l'HMAC, cifra il tutto e trasmette il risultato dell'elaborazione.



I dati sono classificati in quattro stati, due correnti e due pendenti: gli stati di lettura (per i record ricevuti) e scrittura (per l'invio dei record) correnti e gli stati di lettura e scrittura pendenti.

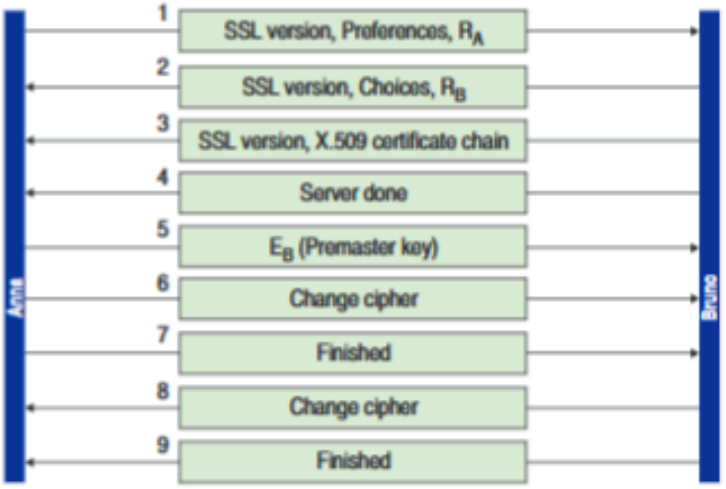
La differenza tra stato corrente e pendente è in funzione delle operazioni che sono state fatte o meno sui dati stessi: quando sono impostati i *parametri* di sicurezza e sono state generate *le chiavi*, il dato passa dallo stato pendente allo stato corrente e viceversa.

I parametri di sicurezza sono impostati fornendo i seguenti valori:
– *connection end*: specifica se l'entità in questione è il client o il server;
– *bulk encryption algorithm*: indica l'algoritmo di cifratura e i relativi parametri, come la lunghezza della chiave;
– *MAC algorithm*: indica l'algoritmo di autenticazione e i parametri relativi;
– *compression algorithm*: indica l'algoritmo di compressione e i relativi parametri;
– *master secret*: sequenza di 48 byte condivisa tra i due interlocutori;
– *client random*: 32 byte casuali forniti dal client;
– *server random*: 32 byte casuali forniti dal server.

Handshake Protocol

L'Handshake Protocol è responsabile della negoziazione dei parametri di sicurezza di una sessione mediante i suoi tre sotto-protocolli, che permettono inoltre di notificare eventuali situazioni di errore.
La fase di "handshake" vera e propria viene iniziata dal client inviando al server un messaggio di "hello" per iniziare la sessione (1) e proponendo la versione del protocollo e la cipher suite: e il server sceglie il protocol e le opzioni presenti nella suite (2): nel caso non si trovasse l'accordo si procede su livelli inferiori fino a trovare una intesa.
I parametri necessari per la comunicazione sono i seguenti:
– *session identifer*: identificatore della sessione scelto dal server;
– *peer certificate*: certificato X.509 dell'interlocutore (può mancare);
– *compression method*: algoritmo di compressione;
– *cipher spec*: algoritmi di cifratura e autenticazione e relativi parametri crittografici;
– *master secret is resumable*: flag che indica se la sessione può essere utilizzata per iniziare nuove connessioni

Si prosegue con la fase di "change cipher spec": il server invia al client un certificato (3-4) attestante la propria identità e contenente la propria chiave pubblica (per esempio una chiave RSA): il client ne verifica l'identità, genera una chiave di sessione casuale (premaster key) e la invia al server, cifrandola con la chiave pubblica (5-6-7).
Ora il server decifra il messaggio con la propria chiave privata e ottiene la chiave di sessione che utilizzerà per cifrare/decifrare il successivo traffico dati con un algoritmo simmetrico (per esempio AES).
Lo schema completo della fase di negoziazione tra *Anna* (client) e *Bruno* (server) è mostrato qui a lato.



Conclusioni

SSL permette la cifratura delle comunicazioni tra client e server ma non garantisce l'identità delle parti: nel web è però fondamentale sapere "con chi si ha a che fare" soprattutto in caso di transazioni con denaro.

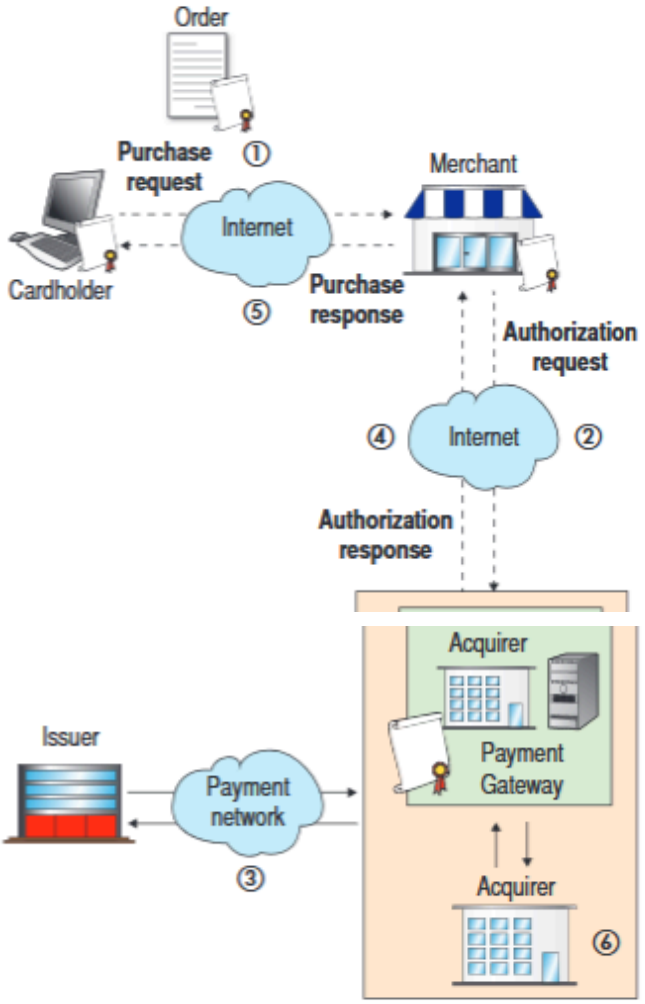
La soluzione è stata sviluppata proprio su iniziativa di Visa e Mastercard per l'utilizzo delle carte di credito: si è realizzato il protocollo SET (Secure Electronic Transaction).

Il SET soddisfa sette importanti requisiti.
1. Fornisce confidenzialità nella trasmissione dei dati di pagamento.
2. Assicura l'integrità di tutti i dati trasmessi.
3. Garantisce che il possessore della carta di credito sia un utente legittimo.

4. Garantisce che il commerciante possa accettare le transazioni attraverso il contatto con una istituzione finanziaria.
5. Assicura l'utilizzo delle migliori pratiche e tecnologie di sicurezza per proteggere ogni legittima parte coinvolta nella transazione di commercio elettronico.
6. Crea un protocollo che non dipende dai meccanismi di trasmissione dei dati.
7. Facilita e incoraggia l'interoperabilità fra software e network provider.

Entrano in gioco nuovi attori, tutti muniti di appositi certificati:

- possessore della carta di credito (cardholder): è il cliente che con la sua carta di pagamento interagisce con il commerciante tramite il personal computer;
- distributore (issuer): istituzione finanziaria che gestisce gli account finanziari per i clienti e distribuisce le carte di credito; deve autorizzare e, quindi, garantire il pagamento;
- commerciante (merchant): colui che offre in vendita beni o servizi e che preventivamente deve aver concordato ed essere accreditato per riceve pagamenti elettronici da un acquirer;
- acquisitore (acquirer): è l'istituzione finanziaria che gestisce l'account del commerciante e processa le autorizzazioni di pagamento e i pagamenti concreti;
- gateway di pagamento (payment gateway): componente fisica che processa le istruzioni di pagamento sia del commerciante sia del cliente; è controllato dall'acquirer o da una designata terza parte.



Seguiamo il flusso delle informazioni dopo che viene effettuato un pagamento mediante SET, come rappresentato nel disegno qui a lato.

1. Il possessore della carta di pagamento (cardholder) invia una richiesta di acquisto al venditore merchant contenente le informazioni sulla merce e sul proprio account.
2. Il venditore contatta il gateway di pagamento e chiede l'autorizzazione al pagamento.
3. Il gateway di pagamento verifica l'autenticità della firma del possessore della carta contattando il distributore del possessore della carta (issuer).
4. Se tutto è corretto il gateway invia una risposta di autorizzazione al venditore.
5. Il venditore consegna la merce al possessore della carta.
6. Dopo aver soddisfatto le richieste del possessore della carta di pagamento, il venditore può richiedere l'accredito, presso il suo acquisitore (acquirer), della somma stabilita.

> ⚠ Le coordinate della carta di credito sono prima codificate con la chiave pubblica dell'issuer poi affiancate ai codici della merce: il numero della carta di credito è fornito in modo crittato e quindi il negoziante non vede il numero in chiaro.

L'unico problema nasce se la carta di credito viene rubata, ma in questo caso il proprietario avvisa subito il proprio issuer che toglie validità al certificato inibendone la validità.