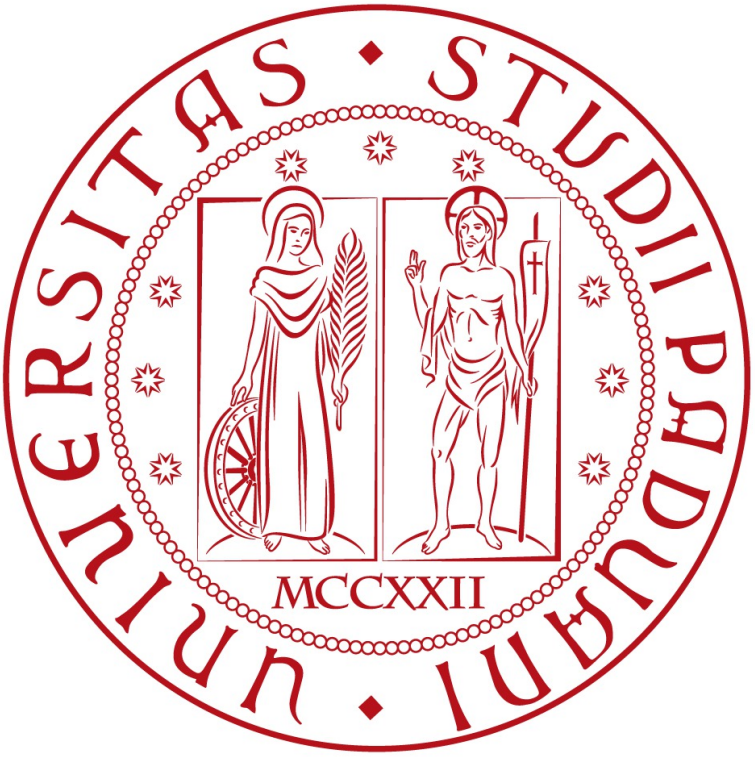
Università degli studi di Padova

Anno accademico 2012-2013

Dipartimento di Ingegneria dell'Informazione

Tesi di laurea triennale in Ingegneria dell'Informazione



Integration of Quantum Key Distribution in the SSL/TLS handshake protocol

Relatore: Nicola Laurenti

Laureando: Nicola Tommasi

Quantum physics seems to be the "magical" form of physics, and its application to cryptography even more magical.

Anonymous

Ai miei genitori e alla mia famiglia, mi hanno sempre sostenuto economicamente e moralmente in questi tre anni.

Allo staff de "La Lumiere", la mia seconda famiglia, un grazie per tutte le serate passate a lavorare e a divertirci insieme.

A Fabio, che mi ha sopportato e supportato negli ultimi tre anni, condividendo con me momenti tristi e felici, sempre pronto a sacrificarsi per gli amici.

Contents

Abstract

Quantum Key Distribution (QKD) is a technology, based on the quantum laws of physics, rather than the assumed computation complexity of mathematical problems, to generate and distribute provably secure cipher keys over unsecured channels. It does this using single photon technology and can detect potential eavesdropping via the quantum bit error rates of the quantum channel. Sending randomly encoded information on single photons produces a shared secret key that is a random string and the probabilistic nature of measuring the photon state provides the basis of its security. Commodity security protocols, such as Transport Layer Security (TLS, often referred to as Secure Sockets Layer or SSL), currently handles the bulk of today's internet encrypted traffic. With the advent of quantum computers, several of the cryptographic constructs underlying the security model of TLS will be broken. Using quantum keying material within this protocol would solve this problem. QKD does offer perfect forward security and therefore past QKD secrets will remain secure, even when the public key algorithm is broken. Although quantum computers are not yet a reality, it is necessary to develop alternative technologies well in advance, since past secrets will be a risk once the threat is realized.

1 Introduzione

La sicurezza è diventata un fattore molto importante per le reti wireless e via cavo. Le caratteristiche di queste reti rappresentano sfide e opportunità per il raggiungimento degli obiettivi di sicurezza, quali la riservatezza, l'autenticazione, l'integrità, disponibilità e il controllo degli accessi. Tecniche crittografiche sono ampiamente utilizzate per le comunicazioni sicure. La maggior parte dei meccanismi crittografici, ad esempio la crittografia simmetrica e asimmetrica, comportano l'uso di chiavi, ovvero di sequenze generate in maniera casuale e note solo agli utenti legittimi. Tuttavia tutte le tecniche risulteranno inefficienti se il metodo di distribuzione della chiave è un meccanismo debole. Lo scopo del meccanismo di distribuzione della chiave è quello di fornire procedure sicure per la gestione di materiali di codifica crittografici. La sicurezza dei più moderni sistemi di crittografia che usano il meccanismo di distribuzione delle chiavi (o Key Distribution) si basa sulla complessità computazionale e sul tempo straordinariamente lungo richiesto per far breccia nel codice. Questo significa che gli attuali sistemi crittografici diventeranno più vulnerabili con la crescita continua delle velocità e delle potenze dei computer. Quantum Key Distribution (QKD) sta attirando molta attenzione come potenziale meccanismo candidato a sostituire i metodi computazionali di distribuzione. Questo perché QKD permette a due parti lontane di generare una chiave segreta, la cui privacy è garantita dalle leggi della fisica quantistica. Così la crittografia quantistica risolve il problema della distribuzione casuale della chiave. Il concetto di crittografia quantistica è stato introdotto da Bennet e Brassard nei primi anni '80, a partire da un'idea precedente di Wiesner. La sicurezza incondizionata del loro protocollo, il BB84, può essere facilmente provata usando il teorema di non-clonazione e quello di non-ortogonalità degli stati, usati per codificare l'informazione. Fino a oggi, diversi protocolli QKD sono stati sviluppati e alcuni di essi permettono di gestire la trasmissione delle chiavi attraverso decine di chilometri sia in fibra, sia nello spazio libero. Così, le reti LAN, presentano un grande interesse per l'uso della QKD, a causa delle aree di coperture limitate di queste ultime.

Vediamo ora una panoramica dei principali algoritmi di codifica usati durante lo scambio di messaggi.

1.1 Algoritmi di Crittazione

Gli algoritmi crittografici vengono usati da SSL/TLS in modalità ESP, per crittare i dati da inviare in modo che la comunicazione avvenga in maniera confidenziale. I principali algoritmi crittografici utilizzati da SSL/TLS sono DES, 3-DES e AES.

- ***DES:*** Il Data Encryption Standard (DES) è un algoritmo crittografico scelto come standard dal National Bureau of Standards (NIST) come Federal Information Processing Standard (FIPS) per il governo degli Stati Uniti d'America nel 1976 e in seguito diventato di utilizzo internazionale. Esso è basato su un algoritmo che utilizza una chiave simmetrica di 56 bit. Proprio per la scarsa lunghezza della chiave, DES è considerato attualmente poco sicuro dalla comunità internazionale.

- ***3-DES:*** Triple Data Encryption Standard (3-DES) è un algoritmo crittografico derivato da DES. Esso consiste nell'applicare tre volte l'algoritmo DES su ogni blocco di dati e utilizza chiavi di 112 o 168 bit.

- ***AES:*** L'Advanced Encryption Standard (AES) è un algoritmo crittografico attualmente utilizzato dal governo degli Stati Uniti. Utilizza chiavi lunghe 128, 192 e 256 bit.

1.2 Diffie-Hellman Public Key Exchange

I principali metodi crittografici prevedono l'uso di una chiave condivisa tra il mittente e il destinatario del messaggio. Se questa chiave cade nelle mani di un intercettatore allora il cifrario diventerà completamente leggibile all'intercettatore come al destinatario. Ma come si possono scambiare le chiavi in modo sicuro dal mittente al destinatario?
Negli anni '60 e '70, quando i computer stavano diventando più potenti, era possibile immaginare milioni, a volte miliardi di transazioni all'anno tra le

aziende. Molte di queste transazioni richiedevano l'uso della crittografia. Anche con l'uso di un cifrario, per ognuna di queste transazioni le parti in causa avrebbero dovuto necessariamente ottenere la chiave in qualche modo e un qualsiasi intercettatore avrebbe potuto trovarla. Whitfield Diffie fu una delle prime persone a escogitare una strategia vincente per affrontare questo problema. Diffie si è laureato al MIT nel 1965 e negli anni seguenti si prese cura del problema appena descritto. Viaggiò attraverso i paesi imparando tutta la teoria sulla crittografia. Nei primi anni '70 incontrò Martin Hellman che faceva parte del dipartimento di Ingegneria Elettrica di Stanford. I due iniziarono a pensare insieme al problema e nel maggio del 1975 Diffie ebbe l'idea fondamentale. Se due persone che non si sono mai incontrate e perciò non hanno una chiave condivisa, vogliono comunicare in modo sicuro allora possono "dividere" in due parti la chiave. Ciascuno avrebbe sia una chiave privata, sia una chiave pubblica. Ad esempio se Alice vuole mandare un messaggio a Bob, lei guarda la chiave pubblica del destinatario e usa quella per criptare il messaggio. Bob ottiene il messaggio e usa la propria chiave privata per decriptarlo. Un intercettatore conosce solo la chiave pubblica ma non quella privata. L'idea è che conoscendo la chiave pubblica deve essere facile criptare il messaggio ma la decrittazione non deve essere facile se non si conosce la chiave privata. Un anno dopo, nel maggio del 1976, Hellman, pensò un modo matematico allettante per implementare l'idea di Diffie per lo scambio della chiave segreta tra le due parti [4].

Supponiamo che Alice voglia comunicare in modo sicuro con Bob ma che non si siano mai incontrati e perciò non hanno una chiave in comune. Il metodo Diffie-Hellman per lo scambio delle chiavi è un metodo che assicura lo scambio della chiave che loro possono usare per codificare i messaggi usando, ad esempio, DES.

- Per prima cosa viene scelto un numero primo molto grande, che chiameremo p. Allora scegliamo $g \in \mathbb{Z}$ in modo che il set $\{g^s | s \in \mathbb{Z}\}$ sia abbastanza grande (Idealmente dev'essere positivo). Dobbiamo tenere in mente che stiamo facendo operazioni in modulo p, quindi g^s sarà un

elemento di $\mathbb{Z}_p$.

- Alice e Bob annunciano pubblicamente p e g. Adesso Alice, sceglie un numero intero casuale $r < p$ e lo tiene segreto. Questo sarà la sua chiave privata. Anche Bob sceglie un numero intero casuale t, la propria chiave privata.

- Alice calcola $X = g^r \in \mathbb{Z}_p$ e Bob calcola $Y = g^t \in \mathbb{Z}_p$. Ora i due si scambiano X e Y usando un canale pubblico accessibile da tutti. Quindi Alice conosce p, g, r, X e Y. Bob conosce p, g, t, X e Y. Per ottenere la chiave condivisa Alice calcola $K = Y^r = g^{tr}$ e Bob calcola $K = X^t = g^{rt}$. Allora Alice e Bob conoscono entrambi K e possono usarlo come chiave.

Consideriamo brevemente cosa conosce l'intercettatore Eve. Alice e Bob annunciano pubblicamente p, g, X e Y. Eve sa anche che $X = g^r$. Se potesse risolvere questa equazione per r, allora potrebbe calcolare $K = Y^r$ per ottenere la chiave segreta. Potrebbe essere tentata di applicare il logaritmo base g ad entrambi i membri e risolvere per r. Ma ricordiamo che per ottenere X, Alice calcola g^r e lo riduce secondo modulo p. Quindi prendendo il logaritmo in base g di entrambi i membri non risolve l'equazione in r che Alice effettivamente usa. Infatti, raramente il risultato per Eve sarà un numero intero. Risolvere $X = g^r$ secondo r è chiamato il problema del logaritmo discreto e si rivela essere difficile fintanto che Alice e Bob scelgono il set $\{g^s | s \in \mathbb{Z}\}$ molto grande. Certamente Eve potrebbe provare tutti i possibili valori di r. Ma Alice e Bob hanno scelto p e g tali che esistono così tante possibilità per Eve di provare che lei e il suo computer non possono fare in un tempo ragionevole. Così, in modo pratico, Alice e Bob hanno generato una chiave segreta condivisa [4].

2 Quantum Key Distribution

Abbiamo appena visto uno dei principali metodi usati per lo scambio delle chiavi tra gli utenti. Questo medoto, il Diffie-Hellman, sembra abbastanza sicuro. In realtà come abbiamo notato si basa su una sicurezza di tipo computazionale ossia per violarlo necessiterebbe di una quantità irragionevole di risorse computazionali che supererebbero di gran lunga il valore delle informazioni. Abbiamo perciò bisogno di una sicurezza più forte, questa ci è data dalle leggi della fisica quantistica. La meccanica quantistica è diventata un concetto standard per la maggior parte di ciò che viene fatto in fisica e anzi ha giocato questo ruolo per tre quarti di secolo. Per tutto questo tempo fisici e filosofi hanno sollevato e discusso questioni riguardo all'interpretazione della meccanica quantistica: Perché abbiamo che una singola misurazione, come un particolare tipo di interazione, causa una risposta probabilistica e irreversibile dalla natura mentre altri tipi di interazione causano cambiamenti deterministici e reversibili? Questa è una domanda interessante e i ricercatori stanno ancora discutendo sulla risposta. Nell'ultimo paio di decadi comunque i ricercatori hanno anche pensato seguendo questa linea: dobbiamo accettare che gli oggetti quantici agiscono in modi contrari all'intuizione e vediamo se possiamo usare questa particolarità tecnologicamente. I due migliori esempi del potenziale uso di queste singolarità quantiche sono la computazione quantistica e la crittografia quantistica. Uno dei primi schemi crittografici quantistici, come già accennato, fu introdotto da Charles Bennet e Gilles Brassard nel 1894. La loro idea era di non usare direttamente il segnale per trasmettere le informazioni segrete. Piuttosto, essi suggerirono di usare segnali quantici per generare una chiave segreta condivisa tra le due parti. Quindi il protocollo Bennet-Brassard è un esempio di "Quantum Key Distribution". Le implementazioni in laboratorio della Quantum Key Distribution usano in genere fotoni per trasportare i segnali quantici. Di solito questi fotoni sono trasmessi su fibre ottiche ma in alcuni casi sono mandati direttamente attraverso l'etere. In principio, una chiave segreta distribuita con la crittografia quantistica può essere usata nei seguenti due modi: Può essere usata una volta sola (one-time pad) oppure come una chiave condivisa

in un algoritmo standard a chiave segreta come DES o AES. Consideriamo brevemente i due casi:

One-Time-Pad: la chiave condivisa potrebbe essere una sequenza di zeri e uni, ad esempio, 011101001000010. Il mittente, Alice, in qualche modo traduce il suo messaggio in cifre binarie; supponiamo che il messaggio tradotto sia 101010101010101. Ora addiziona a ogni bit della chiave al corrispondente bit del messaggio, in modulo 2, per generare il testo cifrato:

```
101010101010101
011101001000010
110111100010111
```

Ora manda questa ultima stringa di bit a Bob, che fa la stessa operazione e ottiene il messaggio originale:

```
110111100010111
011101001000010
101010101010101
```

Nello schema One-Time-Pad, se la chiave è casuale ed è conosciuta davvero solo da Bob e Alice, allora la crittografia è infrangibile. Infatti se per Eve tutte le stringhe binarie sono equiprobabili, allora tutti i testi avranno la stessa probabilità che avevano a priori rendendo così impossibile all'intercettatore ricavare informazioni. Ma è scomodo dover trasmettere nuovi bit della chiave segreta alla stessa velocità dei bit del messaggio inviato. QKD offre una potenziale soluzione al problema e infatti nuovi esperimenti hanno già prodotto tassi di chiave segreta sufficienti per trasmettere un segnale video con la crittografia di tipo One-Time-Pad. Alice e Bob potrebbero non aver bisogno di una crittografia di tipo One-Time-Pad ma solo di una chiave segreta da usare in un crittosistema a chiave segreta. Purtroppo questo crittosistema a chiave pubblicar rischia in futuro di essere smontato dall'avvento drl computer quantistico. I sistemi di QKD non sono vulnerabili all'attacco di un computer quantistico; per questo offrono un vantaggio importante. Inoltre

questo modo di utilizzare la crittografia quantistica può tollerare un tasso più lento di generazione della chiave segreta rispetto a quello necessario per la one-time pad. Comunque la sfida più grande per la crittografia quantistica al momento non è il rate di trasmissione ma la distanza. Esistono prototipi di sistema che connettono terminali distanti diversi chilometri uno dall'altro, ma qualitativamente, nuove sfide sorgono quando si cerca di estendere il sistema per centinaia o migliaia di chilometri.

2.1 Il protocollo BB84

Una misurazione quantica non è un'acquisizione passiva di informazione, è invece una procedura invasiva che tipicamente cambia lo stato delle variabili che vengono misurate. Gli schemi di Bennett e Brassard usano questa caratteristica della misurazione quantica per rilevare l'azione un potenziale intercettatore. Nello specifico possiamo immaginarci la seguente scena. Alice e Bob vogliono generare una chiave casuale e segreta. L'intercettatore, Eve, vuole ottenere alcune informazioni dalla chiave senza essere scoperto. Se ci riesce allora, più tardi sarà in grado di leggere almeno una parte del messaggio segreto codificato con quella chiave. Ora, siccome il protocollo di distribuzione delle chiavi BB84 prevede che Alice invii segnali quantistici a Bob, Eve tipicamente non sarà in grado di misurare questi segnali senza causare qualche disturbo. In questo modo Alice e Bob sperano di rilevare la presenza di Eve e sventare il suo piano. Ma se Eve non può misurare i segnali senza causare disturbo, allora Bob allo stesso modo non può misurare i segnali senza cambiarli. Quindi il protocollo deve essere progettato in modo tale da permetter a Bob di ottenere la chiave corretta, a dispetto del disturbo causato dalle sue misurazioni. Il protocollo di Bennet e Brassard risolve questo problema rendendo possibile ad Alice e Bob di sapere quando Bob ha fatto una misura distruttiva, così possono scartare quei dati e tenere solo quelli che Bob non ha disturbato. Ecco i passi del protocollo BB84:

1. Alice genera due stringhe binarie casuali $A = (a_1, \ldots, a_n)$ e $S = (s_1, \ldots, s_n)$. Gli elementi di A sono zeri e uni; un sottoinsieme di queste entry sarà eventualmente usato per creare una chiave condivisa segreta. Gli ele-

menti di S sono scelte dal set binario $\{+, \times\}$ dove il carattere "+" e il carattere "×" rappresenteranno due diverse basi per lo spazio di stato della polarizzazione dei fotoni.

2. Alice ora manda una sequenza di n fotoni a Bob, la polarizzazione dell'i-esimo fotone viene determinata come segue. Se $s_i = +$, la polarizzazione verrà scelta tra la basi ortogonali $M_+ = (|\updownarrow\rangle, |\leftrightarrow\rangle)$; se $s_i = \times$, la polarizzazione verrà scelta tra le basi alternative $M_\times = ((|\updownarrow\rangle, + |\leftrightarrow\rangle)/\sqrt{2}, (|\updownarrow\rangle, - |\leftrightarrow\rangle)/\sqrt{2}$ i cui elementi formano un angolo di 45 gradi con gli elementi di M_+. In entrambi i casi, Alice usa il primo elemento della base se $a_i = 0$ e il secondo se $a_i = 1$. Possiamo pensare a_i come il bit che Alice sta provando ad inviare a Bob, e s_i come la determinazione del mezzo che vuole codificare con questo bit.

3. Prima di ricevere fotoni, Bob genera una stringa casuale $R = (r_1, \ldots, r_n)$ nella quale ogni entry è scelta dal set $\{+, \times\}$. Quando riceve l'i-esimo fotone, misura la sua polarizzazione nella base segnata con r_i e registra il risultato. Indipendentemente dalla base che usa, se ottiene la prima delle due possibili uscite, registra uno 0 e se ottiene la seconda delle possibili uscite registra un 1. Quindi alla fine della procedura ha una sequenza $B = (b_1, \ldots, b_n)$ di zeri e uni, il risultato delle sue misure. Notiamo il fatto che per ogni fotone dato i, non c'è ragione di assumere che Alice e Bob usino le stesse basi; questo significa che s_i non è necessariamente uguale a r_i. Comunque se hanno usato le stesse basi, questo comporta che per i fotoni non disturbati nel cammino da Alice a Bob dovremmo avere $a_i = b_i$. Per esempio se Alice usa le basi $(|\updownarrow\rangle, |\leftrightarrow\rangle)$ e invia il bit 0, ossia invia a Bob il fotone nello stato $|\updownarrow\rangle$, e Bob misura questo fotone nella stessa base, allora dovrebbe ottenere come risultato $|\updownarrow\rangle$ e registrare il bit 0. Dall'altra parte, se Alice e Bob usano basi differenti, allora non ci sarà correlazione tra i bit di Alice a_i e i bit di Bob b_i.

4. Dopo che tutti i fotoni sono stati inviati, Alice e Bob usano un canale pubblico per scambiarsi le rispettive sequenze di basi usate S e R.

Entrambi comparano queste sequenze e rendono noti i valori degli indici i per cui le due sequenze non coincidono. (Da notare che in questo punto non stanno trasmettendo o comparando nessun valore delle loro stringhe di bit A e B; stanno solo confrontando le basi.) A questo punto Alice rimuove dalla sua stringa di bit A, i bit corrispondenti ai valori degli indici i e Bob fa la stessa cosa con la sua stringa B. Chiamiamo le rimanenti stringhe, più brevi delle originali A' e B'. Di nuovo, assumiamo che non ci siano state interferenze, le stringhe A' e B' dovrebbero essere identiche. La lunghezza di queste stringhe dovrebbe essere circa di $n/2$, poiché per ogni fotone trasmesso c'è la possibilità del 50% che Alice e Bob usino le stesse basi.

Comunicazione Quantistica								
Bit casuali di Alice	0	1	1	0	1	0	0	1
Basi casuali di Alice	+	+	×	+	×	×	×	+
Fotoni polarizzati inviati da Alice	↑	→	↘	↑	↘	↗	↗	→
Basi casuali di Bob	+	×	×	×	+	×	+	+
Bit ricevuti da Bob	0	0	1			0	1	1
Discussione pubblica								
Bob segnala ad Alice le basi utilizzate	+	×	×			×	+	+
Alice segnala a Bob quali erano corrette	ok		ok			ok		ok
Informazione condivisa	0		1			0		1
Raw Key	0		1			0		1

Figure 1: Esempio di comunicazione quantistica tra Alice e Bob

5. In un canale reale è possibile che vi siano errori in una trasmissione anche se non vi sono intercettatori. Così Alice e Bob vogliono ora stimare il numero di errori, che sono, il numero di bit in B' che non sono uguali nella parte opposta A'. Per fare questo Alice invia a Bob, attraverso un canale pubblico, un piccolo campione casuale dei bit in A', che Bob compara con i corrispondenti bit in B'. Dopo il confronto, Alice e Bob scartano questi bit visto che Eve potrebbe conoscerli. Si assuma che i restanti bit abbiano la stessa probabilità d'errore, come quelli controllati, si vuole allora correggere gli errori rimanenti. Alla fine di questo passo Alice e Bob dovrebbero possedere le stringhe A'' e B'', le quali

sono più corte di A' e B' ma almeno sono quasi certamente identiche. Quest'operazione si chiama Information Reconciliation e verrà descritta nel capitolo successivo.

6. Dal numero di errori che Alice e Bob hanno scoperto nel passo 5, possono stimare l'ammontare massimo delle informazioni che un intercettatore potrebbe aver ottenuto dai bit rimanenti. Usano quest'informazione, per rimpiazzare le stringhe A'' e B'' con stringhe più corte A''' e B''' delle quali l'intercettatore non ha alcuna sorta di conoscenza (Privacy Amplification).

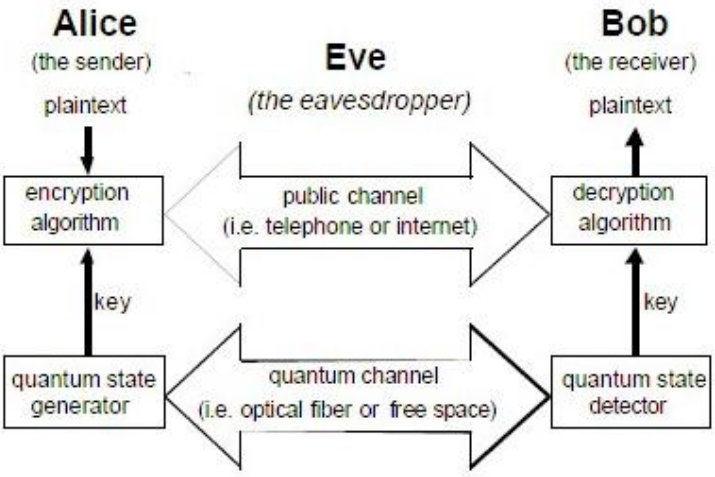


Figure 2: Schema di comunicazione quantistica

2.2 Operazioni di post-processing

Poiché, nel canale quantistico, includendo i dispositivi di trasmissione e ricezione, è sempre presente un certo grado di distorsione e di rumore, anche in assenza di eavesdropper, i bit ricevuti da Bob possono presentare degli errori. Un metodo per risolvere questi problemi consiste nell'applicare due processi chiamati Information Reconciliation e Privacy Amplification. Essi fanno uso di una comunicazione bidirezionale (discussione) tra Alice e Bob, attraverso un canale pubblico.

Information Reconciliation

L'Information Reconciliation è la procedura che permette ad Alice e Bob di correggere le possibili differenze fra i bit inviati da Alice e i corrispondenti bit ricevuti da Bob. Supponendo che Eve sia in grado di ascoltare tutte le comunicazioni pubbliche, Alice e Bob devono riconciliare le proprie chiavi senza rivelare troppa informazione. Ci sono vari algoritmi per questo scopo; qui si descrive il protocollo "Cascade". Essi scelgono una permutazione casuale di posizioni di bit (per rendere casuale la localizzazione degli errori) e dividono la stringhe risultanti in blocchi di b bit, dove b è scelto in modo tale da rendere il più possibile improbabile la presenza di più di un errore per blocco. A questo punto Alice e Bob effettuano un controllo di parità per ogni blocco di bit confrontandosi ogni volta. Nel caso vengano riscontrati errori su un blocco, viene ripetuta iterativamente l'operazione bisecandolo in blocchi più piccoli e effettuando il controllo su di essi fino a trovare l'errore. Per non permettere ad Eve di ricavare informazioni di parità da questo processo, viene tolto il bit finale di ogni blocco nel quale è stato trovato l'errore. Qualsiasi sia il valore di b, è altamente probabile che rimangano ulteriori discrepanze fra i bit che compongono la stringa di Alice e quella di Bob, in conseguenza del fatto che tale protocollo è in grado di correggere un errore per blocco. Per correggere questi errori addizionali, viene ripetuto il processo descritto in precedenza aumentando progressivamente il valore b, fino a quando la probabilità d'errore è sufficientemente bassa. Ora, continuare ad usare sempre questa strategia risulterebbe inefficiente perché forzerebbe Alice e Bob a eliminare almeno un bit per ogni blocco per garantire la privacy della comunicazione. Supponendo ad esempio di avere lasciato esattamente 2 errori e di avere t blocchi, la probabilità che rimangano errori è $1/t$ con un costo di t bit. E' necessario quindi, pensare ad una nuova strategia che diminuisca la probabilità di non rilevare gli errori rimanenti. Si potrebbe infatti considerare che Alice e Bob scelgano un sottoinsieme di posizioni random nelle loro stringhe e che effettuino un controllo di parità sulle sottostringhe formate dai bit presenti in quelle posizioni. Se le stringhe di partenza non sono identiche la probabilità che siano rilevati errori nelle sottostringhe è pari ad 1/2 .

Se questi vengono rilevati, i passi successivi sono simili a quelli della strategia precedente. Viene effettuato infatti iterativamente un controllo di parità bisecando la stringa e eliminando l'ultimo bit, con l'unica differenza che ogni controllo viene svolto su sottostringhe formate con il procedimento appena descritto (ogni sottoinsieme random viene scelto sulle stringhe risultanti dal processo di bisezione). Con questo sistema la probabilità che rimangano errori è pari a 2^{-t} bit, sempre ad un costo di t bit. Una volta che l'ultimo errore viene rilevato, viene ripetuta l'ultima procedura per un certo numero di iterazioni per assicurare che le stringhe possedute da Alice e Bob sono di fatto identiche con una probabilità trascurabile che ulteriori errori non siano stati rilevati.

Privacy Amplification

Dopo aver terminato con successo la fase di Information Reconciliation, Alice e Bob condividono con alta probabilità due stringhe (chiavi) identiche, ma su cui l'avversario ha una quantità di informazione non trascurabile. Ad esempio la presenza di Eve potrebbe non essere stata rilevata in quanto gli errori da lei indotti potrebbero essere stati "coperti" da quelli dovuti al rumore presente sul canale o a imperfezioni della strumentazione utilizzata. Inoltre, anche se durante la fase di Information Reconciliation Eve non recuperasse nessuna informazione riguardo alla stringa inviata da Alice, in quanto l'ultimo bit di ogni sottostringa viene sempre scartato ad ogni controllo di parità, essa potrebbe convertire l'informazione sui bit fisici della stringa in informazione sui bit della chiave finale. Per esempio se Eve conoscesse il bit a della stringa S e Alice e Bob rivelassero il bit di parità di S scartando il bit a, essa conoscerebbe la parità dei bit rimanenti. Se consideriamo che Eve è in grado di ricavare un bit di parità di una stringa quando conosce la parità di un sottoinsieme non vuoto di quella stringa, deduciamo che se Eve è in grado di conoscere al massimo k bit fisici della chiave, essa conoscerà al massimo k bit della stringa dopo la fase di reconciliation. A questo punto Alice e Bob sono in grado di svolgere l'operazione chiamata "Privacy Amplification". Denominiamo S_r la stringa ottenuta dopo l'Information Reconciliation e n la

sua lunghezza. Se Eve conosce al massimo k bit deterministici della stringa S_r, viene scelta pubblicamente una funzione di hash casuale $h(S_r)$ da una classe di funzioni definite di Universal Hashing,

$$h(S_r) : \{0,1\}^n \rightarrow \{0,1\}^{n-k-s}$$

in modo che, scegliendo un valore s opportuno, si diminuisca l'informazione media mutua ricavabile da Eve.

2.3 Considerazioni sull'intercettazione

Uno dei punti di forza del protocollo BB84, o più in generale, di tutti i protocolli utilizzanti la trasmissione di stati quantistici, è l'impossibilità per Eve di "ascoltare" la comunicazione, ovvero di misurare lo stato del fotone e rinviarlo a Bob, senza perturbare statisticamente il sistema e permettere perciò ad Alice e Bob di accorgersi dell'intruso. Alice e Bob possono infatti rivelare alcuni dei bit delle loro stringhe finali, e se questi corrispondono, dedurre che con una buona probabilità nessuno si e frapposto nella loro comunicazione. Infatti seguendo quanto esposto in precedenza, considerando i risultati derivanti dalla meccanica quantistica, possiamo seguire la seguente logica: se non abbiamo nessuna perturbazione o non osserviamo nessun disturbo non possiamo dire con certezza di non aver avuto nessuna intercettazione, possiamo invece dire con certezza che, data una certa soglia n di errori prefissati, se questa soglia viene superata allora il protocollo verrà abortito e la trasmissione ripetuta. Possiamo affermare quindi che il nostro protocollo terminerà sempre in presenza di eavesdropper. Da qui la sicurezza incondizionata.

Questo risulta vero a meno che Eve non applichi questa strategia solamente ad una piccola frazione dei bit trasmessi. In tal caso, l'errore prodotto risulterebbe molto limitato, e diventerebbe quindi arduo per Alice e Bob rilevare la presenza di un intercettatore. Un altro vantaggio è dato dal fatto che per il teorema di no-cloning quantistico, Eve non è in grado di copiare perfettamente il fotone inviato da Alice (non conoscendolo a priori) e misurarlo separatamente una volta che Bob ha annunciato le basi usate.

3 SSL e TLS

3.1 Introduzione

Transport Layer Security (TLS) e il suo predecessore Secure Sockets Layer (SSL) sono dei protocolli crittografici che permettono una comunicazione sicura dal sorgente al destinatario (end-to-end) su reti TCP/IP (come ad esempio Internet) fornendo autenticazione, integrità dei dati e cifratura e operando al di sopra del livello di trasporto. Il protocollo TLS consente alle applicazioni client/server di comunicare attraverso una rete in modo tale da prevenire il 'tampering' (manomissione) dei dati, la falsificazione e l'intercettazione. Nell'utilizzo tipico di un browser da parte di utente finale, l'autenticazione TLS è unilaterale: è il solo server ad autenticarsi presso il client (il client, cioè, conosce l'identità del server, ma non viceversa cioè il client rimane anonimo e non autenticato sul server). L'autenticazione del server è molto utile per il software di navigazione e per l'utente. Il browser valida il certificato del server controllando che la firma digitale dei certificati del server sia valida e riconosciuta da una certificate authority conosciuta utilizzando una cifratura a chiave pubblica. Dopo questa autenticazione il browser indica una connessione sicura mostrando solitamente l'icona di un lucchetto. Questa autenticazione, però, non è sufficiente per garantire che il sito con cui ci si è collegati sia quello richiesto. Per esserne sicuri è necessario analizzare il contenuto del certificato rilasciato e controllarne la catena di certificazione. I siti che intendono ingannare l'utente non possono utilizzare un certificato del sito che vogliono impersonare perché non hanno la possibilità di cifrare in modo valido il certificato, che include l'indirizzo, in modo tale che risulti valido alla destinazione. Solo le CA possono generare certificati validi con un'URL incorporata in modo che il confronto fra l'URL apparente e quella contenuta nel certificato possa fornire un metodo certo per l'identificazione del sito. Molto spesso questo meccanismo non è noto agli utenti di internet ed è causa di varie frodi dovute ad un uso non corretto del browser, non ad una debolezza del protocollo TLS. Il protocollo TLS permette anche un'autenticazione bilaterale, tipicamente utilizzata in applicazioni

aziendali, in cui entrambe le parti si autenticano in modo sicuro scambiandosi i relativi certificati. Questa autenticazione (definita Mutual authentication) richiede che anche il client possieda un proprio certificato digitale, cosa che normalmente non è disponibile in un normale scenario.

3.2 Presentazione dei protocolli

Il "socket layer" si trova tra il livello di applicazione e quello di trasporto nella scala di protocolli TCP/IP. SSL/TLS (o semplicemente solo SSL) contiene due livelli di protocolli. SSL Record Protocol assicura servizi di sicurezza di base a vari protocolli di livello superiore e definisce il formato usato per trasmettere i dati. SSL definisce anche tre protocolli di alto livello che sono usati nel Record Protocol. Questi tre protocolli sono usati nella gestione degli scambi dell SSL. Il primo è il Change Chiper Protocol che carica la suite di codici (una lista di combinazioni di algoritmi crittografici) che viene usata nella connessione SSL. Il secondo è l'Alert Protocol per trasmettere gli avvisi relativi alle entità che stanno comunicando. Il terzo, l'Handshake Protocol, è la parte più complessa del SSL e verrà descritto più avanti [12].

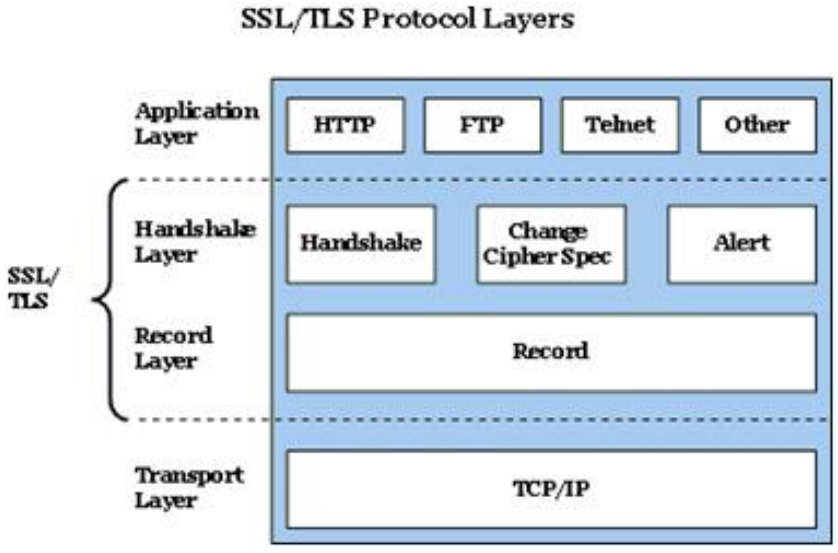


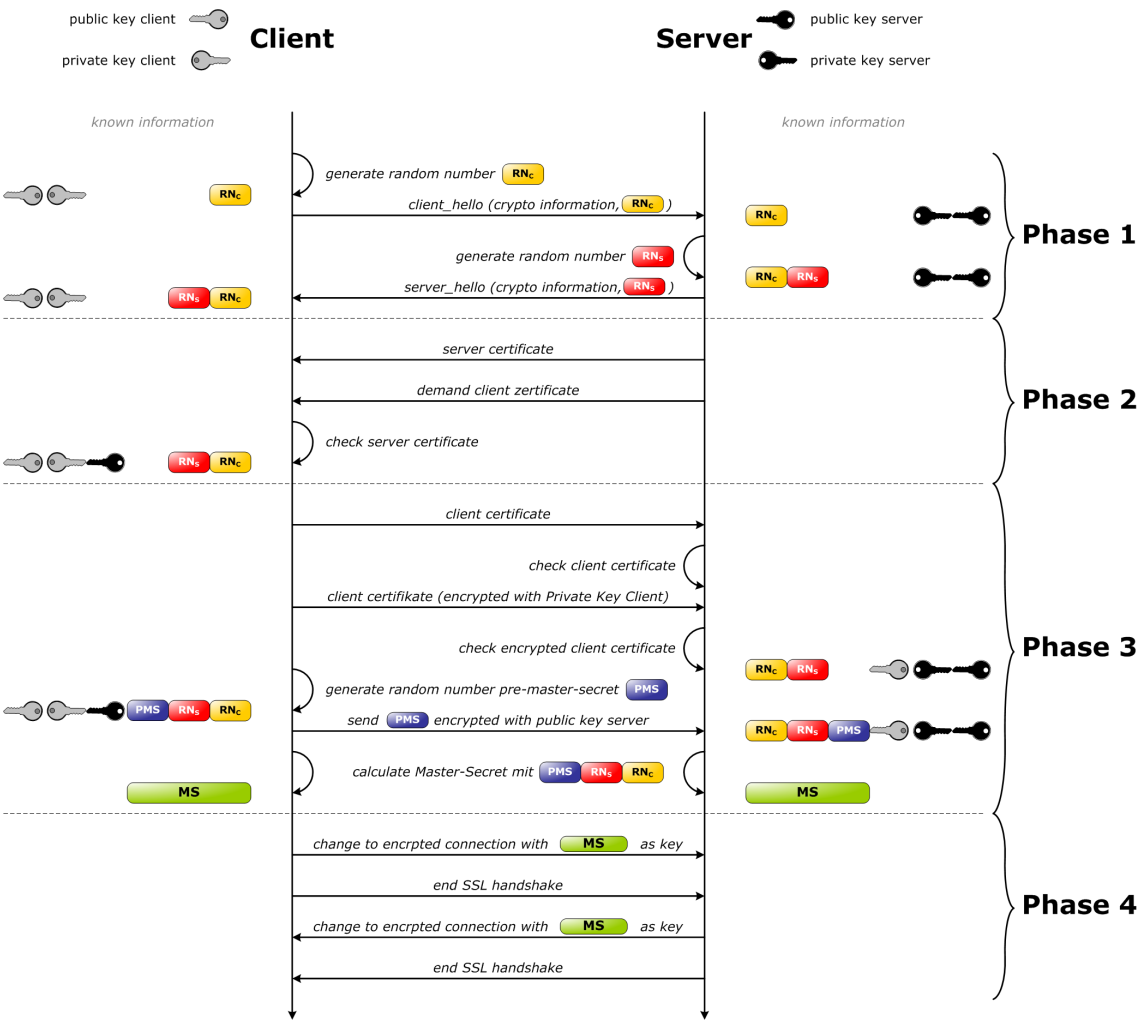
Figure 3: Il protocollo SSL

Una connessione SSL è un trasporto (nella definizione del modello OSI)

che mette a disposizione un servizio di tipo peer-to-peer. Le connessioni SSL sono transitorie e ad ogni connessione è associata una sessione SSL, ovvero un'associazione creata dal protocollo di Handshake. La sessione definisce un set di parametri crittografici sicuri che possono essere condivisi fra connessioni multiple. Una volta che una sessione è stabilita c'è uno stato operativo "corrente" per la scrittura e la lettura. Gli stati "pendenti" di lettura e scrittura sono creati durante l'handshaking. Allora, dopo un handshaking completato con successo, gli stati pendenti diventano correnti (dove per stato si intende stato del protocollo, non stato quantistico). Il Record Protocol fornisce i servizi di confidenza e integrità dei dati per le connessioni SSL. Nella trasmissione il Record Protocol prende i messaggi da trasmettere, frammenta i dati in blocchi più maneggiabili, comprime i dati (opzionale), applica un MAC, codifica e trasmette il risultato in segmenti TCP. Alla ricezione i dati sono decrittati, verificati, decompressi, riassemblati e consegnati agli utenti di alto livello. Quattro tipi di contenuti appartengono al Record Protocol. Questi sono i tre protocolli specifici del SSL (change-chiper-spec, alert e handshake) e l'Application Data, che corrisponde a ogni applicazione che usa SSL. Il protocollo di handshake permette alle parti in comunicazione (client e server) di autenticarsi a vicenda. Permette loro inoltre di negoziare sull'algoritmo di codifica da usare, sul MAC e sulle chiavi crittografiche richieste per proteggere i dati inviati in SSL. Il protocollo consiste in una serie di dati scambiati tra client e server. Lo scambio avviene in quattro fasi [12]

1. ***Stabilire funzionalità di sicurezza:***
   Questa fase è usata per inizializzare una connessione logica e per stabilire le funzionalità di sicurezza associate. Essa parte con un messaggio di client-hello. Durante questa fase il client e il server negoziano sulla versione di SSL da usare, la session ID, il metodo di compressione e la suite di codici.Una suite di codici definisce l'algoritmo per lo scambio delle chiavi e il ChiperSpec che include l'algoritmo di codifica, l'algoritmo di MAC e altre informazioni relative. I metodi di scambio supportati da SSL/TLS sono: RSA, fixed DH (Diffie Hellman), temporary DH e anonymous DH.

2. ***Autenticazione server e scambio delle chiavi:***
   All'inizio di questa fase il server invia il suo certificato (se necessita di essere autenticato). Questo messaggio di certificato è richiesto per ogni scambio di chiave concordato eccetto gli anonimi DH (Diffie Hellman). Successivamente, se richiesto, può essere inviato un messaggio di server-key-exchange. Questo messaggio non è richiesto se come meccanismo di scambio delle chiavi è usato RSA o se il server ha inviato i parametri grazie al fixed DH. Allora, un server non anonimo può richiedere un certificato al client inviando il messaggio di certificate-request. Infine, la fase si conclude con un messaggio di server-done.



3. ***Autenticazione client e scambio delle chiavi:***
   Il client inizia questa fase inviando un messaggio di certificato. Successivamente invia il messaggio client-key-exchange che ha l'obbiettivo di abilitare il client e il server a creare un messaggio di pre-master-secret. Il contenuto del messaggio dipende dal tipo di meccanismo usato per lo scambio delle chiavi. Il pre-master-secret è usato successivamente da entrambe le parti per calcolare il master-secret, un valore di 384 bit che viene generato in ogni sessione. Allora i parametri il Chiper-Spec sono generati dal master-secret usando una certa tecnica di hashing che comprime il tutto. Questi parametri sono un MAC segreto del client, un MAC segreto del server, una chiave segreta del client, una chiave segreta del server, un vettore di inizializzazione IV del client e un altro vettore di inizializzazione IV del server. Infine, il client può inviare un messaggio di verifica del certificato per fornire una verifica esplicita del suo certificato.

4. ***Fine:***
   Il client invia il suo messaggio di change-chiper-spec e copia lo stato pendente nello stato corrente (il messaggio è inviato usando il protocollo di Change Chiper Spec). Allora, manda il messaggio finale codificato dai nuovi algoritmi e chiavi segrete. Infine, il server invia il suo change-chiper-spec, trasmette il pending al current ChiperSpec e invia il suo messaggio finale.

Se esiste una sessione SSL, allora le due parti comunicanti condividono una chiave segreta simmetrica che può essere usata per stabilire una nuova connessione SSL, evitando in tal modo, costose operazioni a chiave pubblica richieste per stabilire la sessione. In due recenti RFC (Request For Comments) sono stati definiti tre nuovi set di chiavi pre-condivise per TLS. Queste PSK (Pre Shared Key) sono chiavi simmetriche condivise in anticipo tra le parti comunicanti. Il primo set di questa suite di codici usa solo operazioni a chiave simmetrica per l'autenticazione. Il secondo set usa come meccanismo di scambio Diffie-Hellman, autenticato grazie a una PSK. Il terzo set combina un'autenticazione a chiave pubblica del server con un'autenticazione di tipo

PSK del client. In realtà, queste suite di codici a chiave precondivisa (PSK) possono essere utilizzate in un solo modo di autenticazione, dove possono offrire autenticazione, protezione e integrità senza riservatezza [13].

4 Integrazione della QKD nel protocollo SSL: QSSL

QSSL o Quantum SSL è la più importante modifica ed estensione del "convenzionale" SSL/TLS. Introduciamo alcuni importanti aspetti di progettazione del protocollo:

1. ***La scelta di SSL/TLS:*** SSL è stato scelto come protocollo base per questo lavoro perché è vastamente usato, relativamente semplice e ben progettato per quanto riguarda la sicurezza.

2. ***Semplicità ed Efficienza:*** Durante lo sviluppo di QSSL, abbiamo provato a introdurre il numero minimo possibile di modifiche ed estensioni a SSL e ne è risultata un'integrazione efficiente di Quantum Computing (QC) in SSL. Questo approccio ha anche abilitato, a scapito della progettazione, un protocollo completamente nuovo che contiene bug sulla sicurezza attesi. In realtà, questa integrazione ha facilitato efficientemente entrambe SSL e QC a ottenere benefici a vicenda. Da una parte, SSL può ottenere nuovi chiave segrete usando QKD, dall'altra, la discussione pubblica richiesta da QC può essere trasmessa usando l'incapsulamento di SSL.

3. ***Generalità:*** QSSL non è diretta verso una specifica implementazione di QKD. Abbiamo provato a fare un'estensione più generale possibile in modo che diverse implementazioni di QKD e varie componenti di fase possano essere considerate. In realtà, altri protocolli di QC, anziché QKD potranno essere considerati in futuro.

4. ***Sicurezza computazionale vs. sicurezza incondizionata:*** Ognuna delle sicurezze di tipo incondizionato usate, One-Time-Pad o la crittografia tradizionale (come DES, 3DES...) ha i propri vantaggi e i propri requisiti. QSSL supporta entrambi i tipi di crittografia mentre SSL/TLS convenzionale non supporta One-Time-Pad.

5. ***Autenticazione dei messaggi:*** Tradizionalmente i MAC usati possono offrire solo la sicurezza computazionale e l'integrità dei dati. I codici di autenticazione basati sull'hashing universale offrono invece sicurezza incondizionata e integrità dei dati. Comunque questi codici di autenticazione hanno bisogno di bit segreti da essere inizialmente condivisi dalle parti autorizzate. Come la QKD può essere usata come sorgente per questi bit, QSSL supporta entrambi i tipi di autenticazione per il traffico dei dati. Questa introduzione del servizio di sicurezza incondizionata e integrità dei dati per il traffico dell'applicazione, potrebbe essere un nuovo importante aspetto per QSSL. Comunque questa funzionalità è esclusa dall'uso obbligatorio dell'autenticazione sicura incondizionata per proteggere le discussioni di QKD sul canale pubblico.

6. ***Inizializzazione di rete:*** Per ottenere sicurezza incondizionata (che è il motivo principale dell'impiego di reti QKD) le reti QKD hanno bisogno inizialmente di chiavi segrete precondivise. Queste chiavi sono richieste per migliorare i primi scambi di sicurezza incondizionata per la QKD sul canale pubblico. Comunque c'è un'altra possibilità di usare la crittografia a chiave pubblica per l'inizializzazione di rete attraverso l'autenticazione solo dei primi passi di QKD. Allora, l'autenticazione sicura incondizionata può essere usata per le successive sessioni di QKD. Nonostante il fatto che questo tipo di autenticazione ibrida e la tecnica di inizializzazione di rete non offrono sicurezza incondizionata in senso stretto, presentano ancora un vantaggio di sicurezza rispetto ad ogni altra tecnica di distribuzione delle chiavi. Perciò, entrambi questi metodi QKD per inizializzare la rete sono supportati da QSSL.

7. ***Applicabilità:*** In un contesto con classiche reti aperte (ad esempio Internet), le reti QKD possono essere considerate come reti "chiuse", specialmente in questa prima fase di sviluppo. Questo è principalmente dovuto alle limitazioni fisiche che esse hanno. Perciò QSSL è più adattabile (almeno per il momento) per applicazioni con intranet privato piuttosto che quelle con Internet pubblico. Comunque, in quanto tali le reti private sono già usate da molte organizzazioni; lo sviluppo di QSSL

potrebbe portare un considerevole vantaggio di sicurezza per questi reti intranet.

L'handshake di QSSL può essere descritto con due modelli. Il modo-1 è basato sulla tradizionale inizializzazione a chiave pubblica di SSL/TLS. Il modo-2 è basato sulle suite di codici a chiave precondivisa, PSK. In caso contrario, è semplice descrivere un singolo modo di QSSL handshake semplicemente etichettando qualche messaggio come situazione dipendente e specificando l'uso della suite di codici.

4.1 Introduzione di un nuovo tipo di contenuto SSL

La prima modifica è l'introduzione di un quinto tipo di contenuto per l'SSL Record Protocol. Questo nuovo tipo viene chiamato crittografia quantistica ed è relativo a qualche protocollo di quantum computing (QC) che deve essere integrato in SSL. Definendo questo tipo di contenuto, il protocollo QC (ad esempio QKD) è considerato come un addizionale protocollo SSL di livello superiore come quelli di Handshake, Change Chiper Spec, e di Alert. Il formato del messaggio del Quantum Cryptography Protocol è mostrato in figura. Ogni messaggio del protocollo ha i seguenti campi:

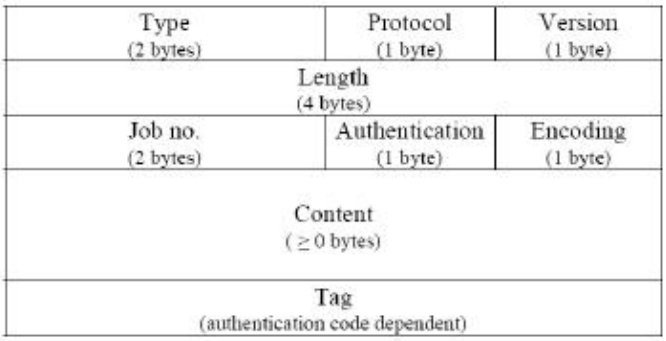


Figure 4: Formato del messaggio del protocollo QSSL

- ***Type*** (2 byte): Indica uno dei messaggi usati nello scambio pubblico del protocollo di QC.

- ***Protocol*** (1 byte): Il protocollo QC usa il protocollo QKD BB84 dovuto a Bennet e Brassard.
- ***Version*** (1 byte): Abilita l'uso di uno o più versioni di un protocollo di QC. Pertanto, componenti di caratteristiche diverse possono essere usati per implementare qualunque fase del protocollo. Per esempio, diverse tecniche di vagliatura, riconciliazione, stima dell'intercettazione (Eve), informazione e/o privacy amplification possono essere usate per implementare il protocollo BB84.
- ***Length*** (4 byte): La lunghezza del messaggio in byte.
- ***Job no.*** (2 byte): Abilita l'operazione di più lavori di protocollo multiplo (o istanze), tutti appartenenti ad un'unica sessione QSSL.
- ***Autentication*** (1 byte): Indica se questo messaggio è autenticato. Può contenere informazioni addizionali circa questa autenticazione.
- ***Encoding*** (1 byte): Indica se una certa tecnica di codifica è utilizzata per il campo "content" del messaggio. Può contenere alcune informazioni ulteriori riguardo questa codifica. Per esempio, una forma di codifica run-length è usata per i messaggi della fase di vagliatura della QKD.
- ***Content*** (≥ 0 byte): I parametri e i dati associati con questo messaggio.
- ***Tag:*** Se il messaggio è autenticato, questo campo conterrà il tag corrispondente all'autenticazione. La sua taglia dipende dal codice di autenticazione usato.

4.2 Definizione di una nuova suite di codici

Vengono definite due nuove suite di codici da usare per QSSL. Il primo di questi set (Modo 1) usa la crittografia a chiave pubblica per inizializzare il processo di QKD. Questo set viene applicato se non ci sono possibilità di autorizzare gli utenti ad avere inizialmente chiavi precondivise richieste per

l'hashing universale. Questo set può essere usato quando gli utenti credono che la sicurezza offerta da tale suite sia adeguata per loro. Il secondo set di codici usa chiavi precondivise per facilitare l'uso dell'autenticazione sicura incondizionata per proteggere il canale pubblico QKD. Questo set fornisce una sicurezza provata e corrisponde al secondo modo di QSSL handshaking (QSSL Modo 2).

4.3 Cryptographic Computation

QSSL usa QKD per generare direttamente i seguenti parametri crittografici: client write MAC secret, server write MAC secret, client write key, server write key, client write IV, server write IV e pre-master-secret. Da notare che viene usata la stessa terminologia dell'SSL/TLS. Comunque, il write MAC secret è utilizzato nella generazione di entrambi i messaggi, tradizionale e incondizionatamente sicuro. Similmente anche i write key sono utilizzati sia per la codifica classica che per quella incondizionatamente sicura. I write IV sono generati solo quando i blocchi di codice tradizionale sono usati per codificare il traffico applicativo. Ogni volta che la codifica e/o l'autenticazione incondizionatamente sicure vengono usate, la taglia delle write key richieste e/o la taglia dei write MAC segreti richiesti deve essere negoziata durante l'handshake del QSSL. La pre-master-secret generata dalla QKD è divisa in due parti. I primi 48 byte compongono la prima parte che possiamo chiamare pre-master-secret-1. I rimanenti bit della pre-master-secret formano la seconda parte, chiamata pre-master-secret-2. Allora, il master secret viene calcolato dalla pre-master-secret-1. La funzione usata per calcolare il master-secret in QSSL è simile a quella usata da SSL/TLS. Successivamente, il master-secret può essere usato per autenticare gli ultimi messaggi di handshake e per chiudere la sessione (quando è permesso). La sessione QSSL può essere chiusa se e solo se entrambi la codifica e gli algoritmi usati per il traffico applicativo non sono incondizionatamente sicuri. Altrimenti, le connessioni non possono essere generate dalle sessioni perché questo può essere fatale per la proprietà specifica di sicurezza incondizionata. La pre-mastersecret-2 è usata come chiave precondivisa per inizializzare le connessioni QSSL future.

Da questo momento, la sua taglia deve essere negoziata dagli utenti durante l'handshake tale che la taglia sia adeguata da abilitare l'uso sella sicurezza incondizionata per proteggere il canale pubblico di QKD. Notare che questa separazione della pre-master-secret in due parti indipendenti è necessaria per la rispettiva natura di sicurezza incondizionata.

4.4 QSSL Mode-1

Questo modo rappresenta l'handshake QSSL basato sull'uso della crittografia a chiave pubblica per l'inizializzazione. Ognuno dei meccanismi di scambio della chiave tradizionali di SSL/TLS (ad eccetto dell'anonimo DH) può essere usato. La sequenza di messaggi scambiata da QSSL nel Modo 1 è molto simile e quella descritta precedentemente nel capitolo 4. Comunque ci sono diverse modifiche da notare. Le più importanti di queste sono:

1. Almeno tre parametri dovrebbero essere aggiunti per la negoziazione nei messaggi di client-hello e server-hello. Questi parametri sono la taglia del write MAC secret, la taglia delle write key e la taglia delle pre-master-secret. La prima di queste è da aggiungere ogni volta che l'autenticazione incondizionatamente sicura è richiesta per il traffico applicativo QSSL. Il secondo parametro è incluso quando si intende usare la codifica One-Time-Pad. Infine, il terzo parametro ogni volta che gli utenti hanno intenzione di generare chiavi precondivise per una futura inizializzazione in Modo 2 (La taglia di questo parametro dovrebbe essere $\geq$ 48 byte). Notare che questo punto viene applicato anche al Modo 2 di QSSL.

2. Il meccanismo di QKD può partire solo dopo che entrambe le parti si sono mutuamente autenticate. Inoltre, QKD deve finire prima dello scambio di qualsiasi messaggio di chiper-spec. Quindi, l'intero scambio del messaggio di QKD è inserito tra la fase 3 e la fase 4 del protocollo di handshake descritto precedentemente.

3. Il meccanismo di QKD continua fino alla completa generazione della taglia della chiave negoziata. Dopo questo punto possono essere scam-

biati solo change-chiper-spec e i messaggi finali (Questo fatto avviene anche per il QSSL).

4.5 QSSL Mode-2

Questo tipo di handshake usa chiavi precondivise (PSK) per l'inizializzazione. Ciò offre un inizializzazione di sessione molto veloce se comparata con la relativa lenta inizializzazione basata su crittografia a chiave pubblica del Modo 1.

Nel Modo 2, lo scambio dei messaggi con QKD è completamente inserito appena prima della trasmissione dei messaggi di change-chiper-spec e quelli finali. Oltre ai tre parametri di sicurezza aggiunti nella negoziazione dei messaggi di client-hello e di server-hello menzionati prima, in questo modo c'è un'importante modifica ai messaggi di server-key-exchange e di client-key-exchange. Nel Modo 2 questi messaggi sono usati per l'identificazione e la sincronizzazione delle chiavi precondivise. All'inizio, il messaggio di server-key-exchange è usato per trasportare una piccola parte di PSK; questa "porzione d'identità" può rappresentare un valore hash di alcuni bit all'inizio della PSK. Questo può essere accompagnato da una sorta di tecnica a finestra scorrevole per scartare alcuni bit non sincronizzati all'inizio dei blocchi di bit. Il messaggio di client-key-exchange, quando ricevuto, è da interpretare come un riconoscimento positivo della "porzione di PSK". Altrimenti, viene mandato dal client un messaggio alert di unknown-psk-identity.

4.6 Gestione delle chiavi

Tipicamente, QSSL può essere considerato come un protocollo che usa QKD per fornire agli utenti chiavi crittografiche istantanee. Questo è verificato a patto che le write key e i write MAC siano usati direttamente dopo la loro generazione per proteggere dati applicativi e il traffico. In questo caso, solo i blocchi di PSK (quando generati) hanno bisogno di un po' di gestione dei singoli termini. Comunque, è possibile usare anche QSSL in modo operativo, o "key-store", dove tutte le chiavi sono generate in taglia più grande e memorizzate per un uso futuro. Questo richiede lo sviluppo della gestione e

dei meccanismi di sincronizzazione per cinque blocchi di chiavi per utente. (un blocco per ognuna delle cinque chiavi generate da QKD). Questo numero duplicherebbe considerando qualche relazione di sicurezza addizionale con un nuovo utente. A questo punto del QSSL è sconsigliato introdurre tali meccanismi. Sarebbe meglio sviluppare questi meccanismi a parte.

4.7 Architettura di sistema

QSSL si trova tra il livello di applicazione e quello di TCP nella scala dei protocolli. Durante la trasmissione, QSSL accetta traffico dati dal livello applicativo e aggiunge la protezione sicura richiesta prima di inviarlo ai livelli sottostanti. Il BB84 è usato come protocollo crittografico quantistico in questa architettura. QSSL usa la rete per inviare due tipi di traffico. Il primo è traffico applicativo, che è codificato e autenticato secondo le richieste dell'utente. Il secondo è il traffico che corrisponde agli scambi sul classico canale pubblico richiesto per implementare il BB84. Quest'ultimo tipo di traffico è trasmesso durante l'handshake e deve essere autenticato in modo da sconfiggere gli attacchi del tipo man-in-the-middle sul protocollo QKD.

La fase di trasmissione quantistica di BB84 avviene utilizzando il canale quantistico. Questa è tecnologia principalmente di livello fisico. Quindi, la gestione e il controllo di questa fase è migliorata ai bassi livelli dell'architettura. Le altre fasi del BB84 (chiamate vagliatura, riconciliazione, stima dell'informazione di Eve, e privacy amplification) usano il canale pubblico classico per le discussioni richieste. Queste fasi sono implementate usando uno specifico modulo di applicazione in modalità utente chiamato protocollo motore del BB84. L'input di questo motore è una riga di bit quantici risultanti dalla trasmissione quantistica, mentre l'output sono bit segreti che sono (temporaneamente) memorizzati in un magazzino per le chiavi e nel modulo gestionale. Questi bit sono usati da QSSL come chiavi crittografiche in accordo con le specifiche suite di codici. Una versione semplificata di QSSL è stata implementata su tecnologia Microsoft "WinSock" usando Visual C/C++ v.6 e con sistema operativo Windows XP. Questa implementazione di QSSL è stata integrata con il nostro protocollo motore BB84 che abbiamo sviluppato prece-

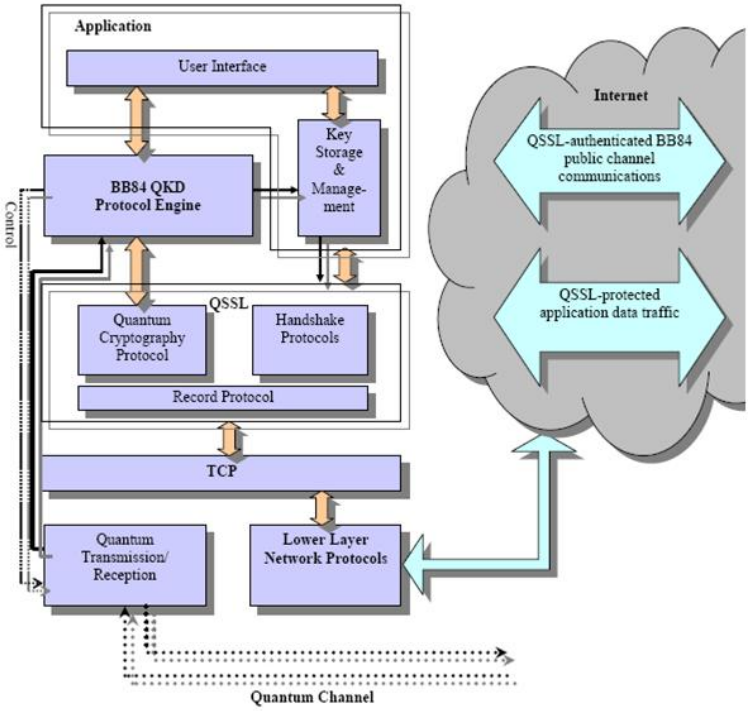


Figure 5: Architettura del protocollo QSSL

dentemente. Questo protocollo motore è basato sulla simulazione software del livello fisico di BB84. Include anche del codice adattabile per migliorare altre fasi di BB84. L'autenticazione sicura incondizionata richiesta per le comunicazioni sul canale pubblico di BB84 può essere realizzata rispettando le nostre versioni precedenti e autenticate di BB84. Questa implementazione autenticata di BB84 usa il codice di autenticazione di Taylor per ottenere tag autenticativi incondizionatamente sicuri. L'avvio sperimentale consiste di due istanze di QSSL installate su due PC (ognuno con processore Intel Pentium IV da 1.7 GHz). Questi due PC sono stati connessi via Ethernet. I risultati del campione mostrano l'importo netto di guadagno dell'espansione della chiave e sono illustrati nelle figure 6 e 7.

In queste due figure tutte le sessioni QKD sono state migliorate del 5% per quanto riguarda il tasso di errore sul bit quantico (QBER). Inoltre, il 20% dei bit vagliati viene usato per un confronto pubblico per stimare l'importo di QBER. Questa stima è richiesta per settare i parametri della fase di riconcili-

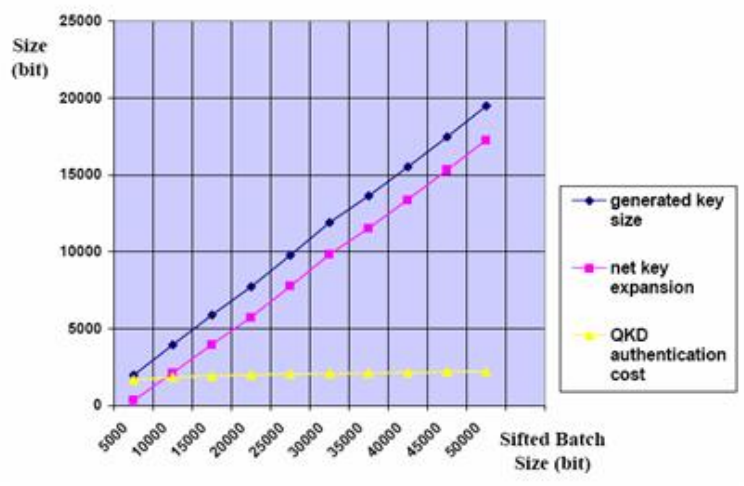


Figure 6: Guadagno netto di espansione della chiave per le sessioni QSSL di modo 2

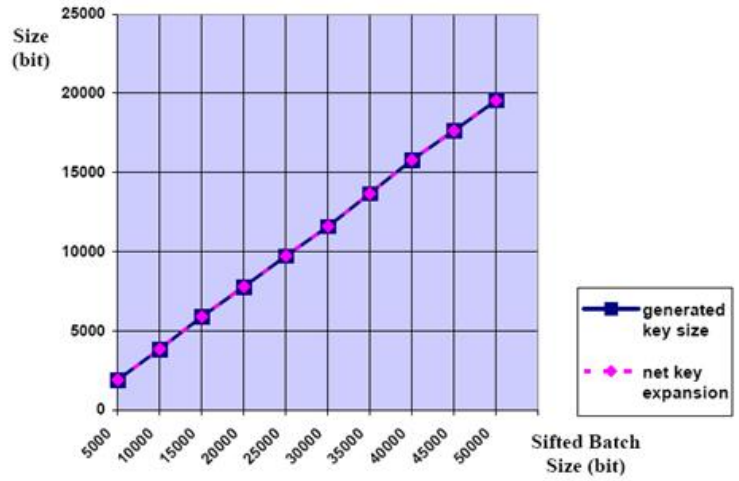


Figure 7: Guadagno netto di espansione della chiave per le sessioni QSSL di modo 1

azione e per decidere se procedere con il protocollo QKD o no. L'informazione sulle chiavi crittografiche generate dalle sessioni QKD è stata ben ridotta sotto i 10^{-70} bit usando la tecnica della privacy amplification. Entrambe le figure 6 e 7 mostrano i risultati dell'espansione delle chiavi per diverse taglie di stringhe di bit vagliate che vanno dai 5000 ai 50000 bit. La prima figura mostra risultati tipici ottenuti da sessioni QSSL di Modo 2, laddove sono

stati utilizzati tag identificativi per proteggere gli scambi di canale pubblico. La curva del costo di autenticazione in questa figura rappresenta l'importo di bit di PSK richiesti per la continua autenticazione incondizionatamente sicura di tutte le discussioni di canale pubblico rilevanti. La curva di espansione di guadagno della chiave è stata prodotta estraendo l'importo del costo di autenticazione dal corrispondente valore della taglia della chiave generata. E' ovvio che a dispetto del nome, QKD è una tecnica di espansione della chiave (o di crescita della chiave) piuttosto che di distribuzione.

La Figura 7 mostra risultati tipici di corrispondenti sessioni QSSL di Modo 1. In quest'ultimo caso è stata utilizzata una tecnica tradizionale di autenticazione con messaggio SSL/TLS per implementare l'autenticazione della discussione QKD su canale pubblico. Quindi, il costo di autenticazione per questo modo è nullo. Inoltre l'importo dell'espansione delle chiavi per le sessioni di Modo 1 è più grande di quello ottenuto dalle corrispondenti sessioni di Modo 2. Comunque le chiavi prodotte dalle sessioni di Modo 1 non hanno la proprietà di sicurezza incondizionata come quelle risultanti dalle sessioni di Modo 2. Infine è importante notare il costo dell'autenticazione incondizionatamente sicura dei messaggi del canale pubblico può essere considerata minore di quella mostrata in Figura 6 Questo avviene quando viene usata l'autenticazione "counter-based". In tutti i casi è meglio usare grosse taglie di bit da vagliare in modo da ottenere un miglior guadagno di espansione della chiave.

È ben giustificato e prudente ora ottenere servizi di sicurezza incondizionata basati combinando la QKD con la one-time-pad o con codici di autenticazione incondizionatamente sicuri. Tuttavia, capire il valore di tali servizi richiede sforzi di ricerca multidisciplinare. QSSL è un passo utile ad una migliore comprensione del requisito dell'integrazione della Quantum Crypthography nelle infrastrutture già esistenti e ben testate.

5 Integrazione della QKD nel protocollo TLS: QKD-TLS

5.1 Perchè l'integrazione della QKD nel protocollo TLS?

Il TLS Handshake Protocol provvede, attraverso un processo di gestione delle chiavi a fornire i parametri di sicurezza per garantire una comunicazione sicura tra gli host. Nel protocollo TLS la gestione delle chiavi è limitata solo ai meccanismi di scambio RSA e Diffie-Hellman; Nessuno di essi è però incondizionatamente sicuro e/o non può essere attaccato. La loro sicurezza dipende dalla potenza computazionale o dal tempo (tempo di esecuzione). È provato scientificamente che QKD è un meccanismo incondizionatamente sicuro e nel suo caso la sicurezza è raggiunta indipendentemente dalla potenza dell'intercettatore. Per questa ragione viene proposto di integrare QKD nel protocollo TLS invece che DH o RSA per lo scambio delle chiavi.

5.2 Requisisti di QKD-TLS

Per integrare QKD nel protocollo TLS devono essere soddisfatti alcuni requisiti:

a) ***Un canale ottico:*** la crittografia quantistica usa fotoni per codificare informazioni. Attualmente ci sono due mezzi per trasportare fotoni: la fibra ottica o lo spazio libero. Recenti lavori di ricerca hanno sperimentato l'uso di atomi e di elettroni come particelle quantiche [15] e in futuro un nuovo tipo di canale quantico potrà essere sviluppato.

b) ***Un modem ottico:*** l'obiettivo del modem ottico è di inviare e rilevare i fotoni. Il modem deve includere un rilevatore di fotoni e un emittente e polarizzatore per singoli fotoni per codificare i dati. E' usato per ottenere la chiave quantica ma può anche essere usato per scambiare dati a seconda del motodo di codifica usato. Per raggiungere la sicurezza incondizionata è richiesto un protocollo a chiave quantistica e deve essere implementato tra due modem ottici. Il protocollo a chiave quantistica

fornirà una chiave sicura che sarà memorizzata in una memoria flash in modo da essere usata quando i dati verranno codificati.

5.3 QKD Configuration Protocol

Nell'obiettivo di garantire un nuovo schema di TLS (incluso il servizio di QKD), è stato introdotto un componente addizionale , il QKD Configuration Protocol che gioca il ruolo di configurazione della sottorete QKD. Quindi il nuovo protocollo TLS avrà cinque componenti : i classici protocolli handshake, Change Spec, Alert, Application e in aggiunta il QKD Configuration. Il formato del messaggio del protocollo di configurazione QKD contiene, oltre ai campi già visti nel QSSL (Type, Protocol, Version, Length, Authentication, Encoding, Content, Tag) i campi di:

- ***Key-Length:*** (4 byte): mostra la lunghezza della chiave ottenuta dal protocollo di QKD. La sua lunghezza varia da 1 a 4 byte. Scegliamo la lunghezza molto grande in modo da usare one-time-pad per ottenere sicurezza incondizionata. In questo caso la lunghezza della chiave deve essere uguale alla lunghezza dei dati da codificare [16].

- ***TTL field:*** (2 byte, minimo un bit): mostra, in termini di tempo (in secondi) o il numero di messaggi in cui una chiave potrebbe essere usata nel processo di codifica. Una volta che il tempo è finito o il numero massimo di messaggi è stato raggiunto, il meccanismo di QKD provvede a generare una nuova chiave.

- ***T field*** (1 bit): questo campo specifica se usiamo il tempo o il numero di messaggi in cui la chiave può essere usata nel processo di codifica. Quando il valore è 1, il campo TTL mostra l'ammontare di tempo, quando invece il valore è 0, il campo TTL corrisponde al numero di messaggi.

5.4 Quantum TLS Handshake Protocol

Nel nostro schema di protocollo TLS , abbiamo introdotto anche alcune modifiche al protocollo di Handshake. In nostro obiettivo è di generare parametri sicuri grazie al servizio di QKD. Nel protocollo di handshake abbiamo sostituito la procedura del processo classico di scambio delle chiavi con il meccanismo di QKD usando il protocollo BB84, lo chiameremo perciò Quantum TLS Handshake. La figura riassume come diversi messaggi siano scambiati tra il client e il server durante l'handshake tra i due.

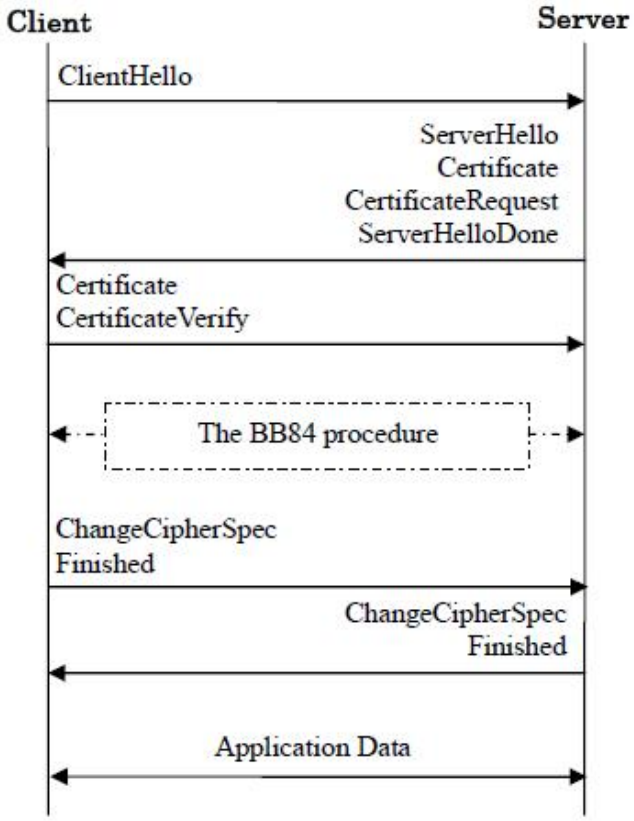


Figure 8: Flusso di messaggi nel quantum TLS handshake

La procedura BB84 integrata in TLS Handshake è mostrata in figura. Siccome BB84 è vulnerabile agli attacchi di tipo man-in-the-middle, verifichiamo, una volta finita l'esecuzione del processo se l'autenticazione è andata a buon fine, dal calcolo del messaggio finale sia del client che del server.

All'inizio della fase di trasmissione quantistica, il client invia al server una serie di fotoni polarizzati. Il numero di fotoni che deve essere inviato dipende dalla lunghezza della chiave desiderata, dall'algoritmo di correzione degli errori, e dall'algoritmo di privacy amplification usato. Per ogni fotone, il server sceglie casualmente uno stato a_{ij} e lo invia al client. I rimanenti passi nella discussione pubblica(fasi di correzione degli errori e di privacy amplification) sono esattamente gli stessi descritti nel protocollo BB84.

5.5 TLS Protocol in Operation Mode

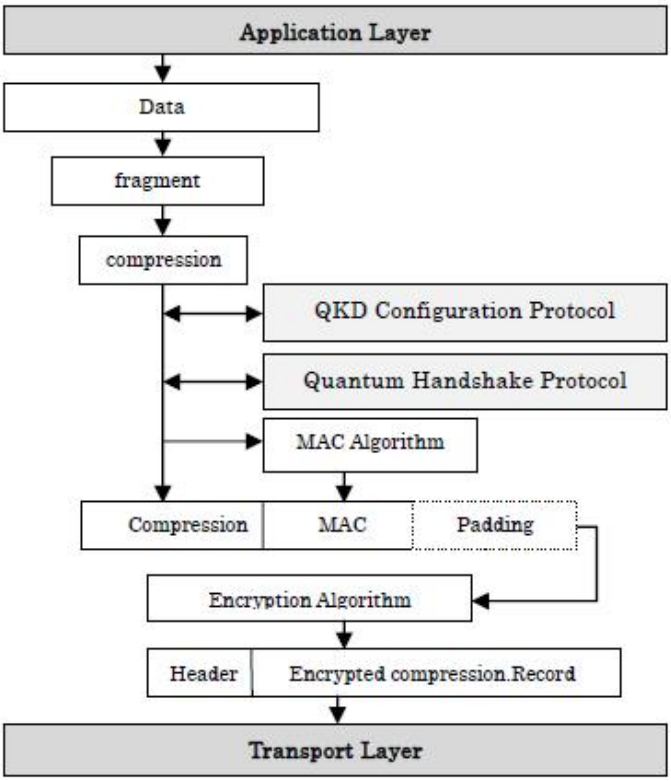


Figure 9: La funzionalità del nuovo protocollo TLS

Il nostro obiettivo principale è di usare QKD per stabilire la pre_master_secret presentata con la formula di calcolo della master_secret, la quale è usata nel calcolo dei messaggi di chiusura di TLS e per generare il materiale chiave per la codifica dei dati nel protocollo TLS. Quindi, QKD è sfruttata nella

procedura di autenticazione e di codifica dei dati tra il client e il server. Il protocollo TLS nel nostro sistema ha bisogno di due modifiche, come in Figura 9.

Per prima cosa dobbiamo integrare un protocollo chiamato QKD Configuration Protocol e dobbiamo modificare l'originale TLS handshake protocol (Quantum TLS Handshake Protocol) per includere il servizio di QKD.

Nel modo operativo, quando il TLS Record riceve dati dal livello applicativo, il QKD Configuration Protocol viene scambiato tra il client e il server per accordarsi sulla lunghezza desiderata della chiave e sui campi mostrati prima. Quando il QKD Configuration Protocol viene eseguito, inizia una nuova sessione di Quantum TLS Handshake. Una volta finita l'autenticazione mutua, definita dallo scambio di certificati, il client e il server iniziano il protocollo BB84 per stabilire la pre_master_secret. Il server corrisponde ad Alice ed il client a Bob perché l'autenticazione mutua (prima della procedura BB84) deve aver successo, dalla verifica di entrambi i certificati del client e del server e l'ultimo certificato verificato è quello del client. Come menzionato prima, il protocollo BB84 è vulnerabile all'attacco man-in-the-middle, così per verificare che il processo di autenticazione è stato stabilito correttamente, il client e il server devono calcolare il messaggio finale TLS usando la chiave segreta condivisa generata dal meccanismo di QKD, $K = \{0,1\}^n$. Abbiamo:

$$pre\_master\_secret = K$$

Il messaggio finale TLS è calcolato dall'espressione:

$$PRF(\ master\_secret, finised\_label,\ hash(handshake\_messages))$$

Deduciamo che il calcolo dei messaggi finali di TLS usano la chiave generata da QKD perché abbiamo:

$$master\_secret = PRF(pre\_master\_secret,$$

"master_secret", Clienthello.random+ServerHello.random)

E' molto importante notare che tutti gli scambi di messaggi pubblici durante la procedura di BB84 sono parte del valore di *handshake_messages*. Quando il server riceve il messaggio finale TLS dal client, calcola il proprio messaggio finale TLS e verifica se è lo stesso del client oppure no; se si, allora il client è autenticato con successo. La stessa operazione viene fatta dal client quando riceve il messaggio finale del server. Così concludiamo che il meccanismo di QKD è d'aiuto per verificare l'autenticità del server. Inoltre, il Record Protocol, richiede un algoritmo per generare le chiavi richieste dallo stato corrente di connessione dai parametri sicuri forniti dal protocollo di handshake. Il master secret è ampliato in una sequanza di byte sicuri che sono a loro volta divisi in un client MAC key, un server MAC write key, un client write encryption key e un server write encryption key. Quindi concludiamo anche che il meccanismo di QKD assicura la sicurezza della codifica dei dati nel protocollo TLS.

5.6 Esempio di applicazione di QKD-TLS

Per dimostrare la fattibilità funzionale, presentiamo in questa sezione un esempio di implementazione di QKD-TLS. Consideriamo due reti LAN connesse da due modem ottici che giocano il ruolo di canale quantico e classico allo stesso tempo. Concentriamoci sugli specifici nodi di connessione C e D come mostrato in figura .

Per assicurare la sicurezza della connessione TLS tra C e D usando il meccanismo di QKD devono essere eseguite cinque fasi:

- ***Fase 1:*** quando il TLS Record Protocol del nodo C, per esempio, prende i dati dal livello di applicazione, chiama il suo Application Data Protocol. Così i dati sono frammentati e per ogni frammento viene fatta una compressione.

- ***Fase 2:*** il TLS record Protocol chiama il QKD Configuration Protocol in modo che i nodi C e D si mettano d'accordo sui parametri del QKD

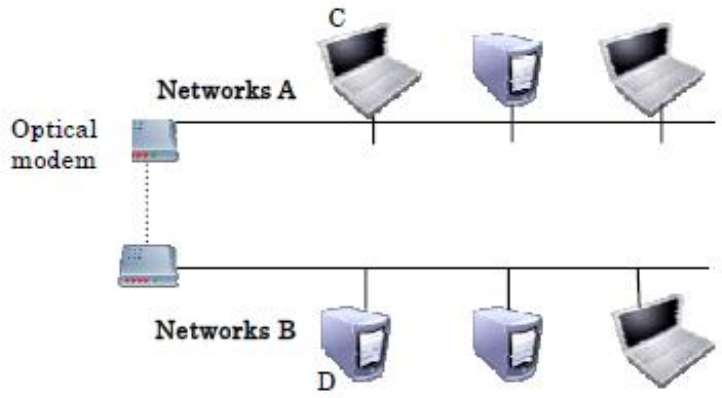


Figure 10: Esempio d'uso di due modem ottici

Configuration Protocol format illustrati in precedenza. I campi più importanti sono: $protocol, key - length$ e i campi TTL e T. Nell'esempio scegliamo il protocollo BB84 perché è il primo protocollo sperimentato ed è provata la sua sicurezza incondizionata. Supponiamo inoltre che la versione sia $version = 1$. Assumiamo che non vi siano meccanismi di autenticazione e di crittazione. Proponiamo queste scelte: $Key - length = 20$ $byte$, $TTL = 200$ $messaggi$, $T = 0$. Se pianifichiamo di usare one-time-pad, per testare la sicurezza incondizionata dobbiamo scegliere $TTL = 1$ $messaggio$.

- ***Fase 3:*** Il TLS Record Protocol usa il Quantum Handshake Protocol per ottenere i parametri di sicurezza. Così il Quantum Handshake Protocol inizia e durante la procedura BB84 viene implementato l'ononimo protocollo nei due modem. La chiave generata viene memorizzata in una memoria flash per essere usata più tardi nella codifica dal Record Protocol.

- ***Fase 4:*** Il TLS Record Protocol riceve la chiave fornita dal servizio QKD e costruisce i suoi parametri di sicurezza.Questi parametri sono usati per generare chiavi per codificare dati e assicurare l'integrità (MAC) come illustrato in figura 10.

- ***Fase 5:*** una volta che il record (frammenti compressi, MAC e pudding

opzionale) è stato crittato, al blocco appena codificato viene aggiunto un header e l'intero pacchetto viene passato al livello di trasporto.

6 Conclusioni

Confrontando alcuni articoli abbiamo presentato dei modelli per implementare la QKD nei protocolli di handshake di SSL e TLS. È stata mostrata inizialmente la differenza tra sicurezza computazionale e sicurezza incondizionata. La prima, seppur ancora largamente usata, basa la propria sicurezza su problemi matematici che ancora non hanno trovato soluzione, ad esempio quello del logaritmo discreto o la fattorizzazione in numeri primi di un numero molto elevato. Questi "problemi" potrebbero in realtà venire risolti grazie alla realizzazione del computer quantistico. Il computer quantistico potrebbe smontare in un attimo tutti gli algoritmi che basano la loro sicurezza su un problema di tipo computazionale. Per fortuna questo computer è ancora in via di sperimentazione e per il momento il suo sviluppo sembra essersi fermato a causa degli elevati costi per il mantenimento delle informazioni. Non è escluso comunque che nei prossimi anni i ricercatori riescano a trovare soluzioni più economiche ed efficienti; Per questo molte aziende hanno già iniziato a correre ai ripari, sfruttando la sicurezza incondizionata della QKD. Questo metodo di distribuzione delle chiavi basa la propria sicurezza su leggi della fisica quantistica e permette di sapere con certezza se e quante informazioni sono state intercettate durante lo scambio della chiave.

In particolare, nell'articolo [2] è stato presentato il Quantum TLS Handshake per assicurare la sicurezza del TLS handshake; il meccanismo delle chiavi è stato fatto da QKD invece che dal classico scambio come RSA o Diffie-Hellman. Inoltre, abbiamo aggiunto un nuovo componente al TLS Protocol in modo da rendere il nostro schema applicabile. Dal nuovo schema abbiamo ottenuto i seguenti vantaggi:

1. Lo scambio di messaggi durante il Quantum Handshake Protocol, è diventato più semplice: il certificato del server non deve contenere parametri pubblici riguardanti lo scambio delle chiavi e non abbiamo bisogno di inviare o di ricevere messaggi dello scambio delle chiavi classico.

2. L'implementazione del nostro schema non richiede di costruire nuovi

dispositivi quantici. Il modem ottico è composto da componenti standard già esistenti come rivelatore di fotoni e sorgente di singoli fotoni.

3. La sicurezza incondizionata può essere raggiunta a un costo davvero basso. Molte organizzazioni e compagnie stanno già usando la fibra ottica. Pertanto, le organizzazioni possono usare le già esistenti infrastrutture per generare chiavi grazie al servizio della QKD.

Infine, dato che nel Quantum Handshake Protocol l'autenticazione è assicurata dalla scambio di certificati, l'autenticazione non è incondizionatamente sicura. Per questo motivo, i ricercatori stanno lavorando per rendere TLS incondizionatamente sicuro, usando autenticazioni incondizionatamente sicure come la Wegman-Carter [17].

Abbiamo visto quindi che uno dei più grandi benefici della QKD è quello di generare e distribuire chiavi sicure attraverso canali non sicuri. Ci sono però ancora delle limitazioni, riscontrate in questa tesi, come il costo della tecnologia o il fatto che la QKD richiede un hardware dedicato; Il suo sviluppo è ancora agli inizi e il tasso di generazione delle chiavi decresce con l'aumentare della distanza. Certamente la distanza tra due terminali potrebbe essere coperta da piccoli sistemi intermedi ma, come visto nei primi capitoli, tutti questi nodi richiederebbero una singola verifica perchè la strategia vada a buon fine. L'ingegneria e la produzione di massa potrebbero rendere l'hardware più economico e ricerche future potrebbero risolvere tutte le limitazioni attuali.

7 References

[1] S.T.Faraj *Integrating Quantum Cryptography Into SSL*, UbiCC Journal, Volume 5

[2] M.Elboukhari, M.Azizi, A.Azizi *Integration of Quantum Key Distribution in the TLS Protocol* IJCSNS, VOL.9, No.12, Dicembre 2009

[3] A.Milk, S.Frankel, R.Perlner *Quantum Key Distribution (QKD) and Commodity Security Protocols: Introduction and Integration* IJNSA,Vol 1, No.2, Giugno 2009

[4] S.Loepp, W.K.Wootters *Protecting Information: From Classical Error Correction to Quantum Cryptography*, Cambridge University Press, Luglio 2006

[5] C.H.Bennett and G.Brassard, *Quantum Cryptography: Public key distribution and coin tossing* in Proc.Int.Conf.Comput.SysSignal Process, Bangalore, India, 1984, pp.175-179

[6] T.Dierks, E. Rescorla, *The Trasport Layer Security (TLS) Protocol*, Version 1.2, RFC 5246, Agosto 2008

[7] R.Alleaume Ed.: *SECOQC white paper on quantum key distribution and cryptography*, Secoqc-WP-v5, Version 5.1 (2007)

[8] R.Canetti: *Universally composable security: A new paradigm for cryptography protocols*, Proceeding of FOCS'01, pp 136-145 (2001)

[9] R. Hughes, Ed: *A quantum information science and technology roadmap*, Seconda parte,(2004)

[10] C.Elliot, *Building the quantum network, New Journal of Physics*

[11] V. Fernandez et al: *Passive optical network approach to gigahertz-clocked multiuser quantum key distribution*, IEEE Journal of Quantum Electronics, Vol. 43, No. 2, (2007)

[12] W. Stallings: *Cryptography and Network Security*, 3rd Edition, Pearson Education International, USA (2003).

[13] P. Eronen and H. Tschofeing, Eds.: *Preshared key ciphersuites for TLS*, RFC 4279 (2005)

[14] N. Ferguson and B. Schneier: *A cryptographic evaluation of IPSec*, Counterpane Internet Security Inc. (1999)

[15] P.Knight *Manipulating cold atoms for quantum information processing*, QUPON conference, Vienna, 2005.

[16] Shannon, C.E *Communication theory of secrecy systems*, Bell System Technical Journal 28-4. URL: http://www.cs.ucla.edu/jkong/research/security/shannon.ht ml (1949)

[17] M.N. Wegman, and J.L. Carter, *New hash function and their use in authentication and set equality*, Journal of Computer and System Sciences, Vol. 22, pp. 265-279, (1981)