Università degli Studi di Trento
Corso di Laurea Specialistica in Ingegneria delle Telecomunicazioni – AA 2007/08
Wireless Networks

# RELAZIONE DI LABORATORIO

Laboratorio #4
VERIFICA DELLA SICUREZZA DI UNA TRASMISSIONE 802.11 PROTETTA CON PROTOCOLLO WPA MEDIANTE ATTACCO BRUTE-FORCE

Gruppo A5
Davide Perina (128698)
Nicola Speri (129144)
Stefano Testi (128697)
Michele Vincenzi (128689)

## 1. INTRODUZIONE

La protezione dei dati è un aspetto critico nelle trasmissioni wireless, dal momento che chiunque si trovi all'interno del raggio di copertura dei dispositivi appartenenti ad un *BSS* o ad un *IBSS* è potenzialmente in grado di intercettare i frame trasmessi, leggerne il contenuto e quindi avere accesso all'intera comunicazione. Per proteggere da eventuali intercettazioni sono stati sviluppati diversi protocolli di protezione dei dati. Con questa relazione si intende verificare (a puro scopo didattico e su dispositivi di test) il livello di affidabilità di un link wireless protetto mediante *WPA* a chiave condivisa (*WPA pre-shared key*). L'analisi di tale schema di protezione è indipendente dall'adozione all'interno di un *BSS* o di un *IBSS*, pertanto lo studio e i risultati ottenuti sono validi per entrambi i casi, e si farà riferimento per semplicità ad un classico *BSS*.
Il metodo di analisi utilizzato consiste nella cattura dei messaggi relativi al processo di autenticazione di una nuova stazione alla rete e nella successiva ricerca offline della parola chiave mediante un attacco a dizionario.
In questo caso il livello di utilizzo del canale (come pure la presenza di altri *AP* su canali adiacenti) non influisce sul risultato dell'esperimento, dal momento che, una volta catturati i 4 frame relativi all'autenticazione, tutto il processo di analisi avviene in modalità offline.

## 2. SETUP SPERIMENTALE

Per l'esecuzione dell'esperimento sono stati utilizzati un *Access Point* e due *laptop*, le cui caratteristiche vengono riportate in tabella 1 e 2 e la cui disposizione viene indicata in figura 1.

| | |
|---|---|
| *AP* | Cisco Aironet 1310 |
| *FIRMWARE* | 12.3(8)JEA3 |
| *FUNZIONI SUPPORTATE DAL FIRMWARE* | Multiple SSID (fino a 16), per ogni SSID: VLAN, authenticazione via MAC address, 802.1x, WPA, numero massimo di stazioni fissabile |

*Tabella 1*

| | |
|---|---|
| *MODELLO* | Dell Latitude 110l |
| *OS* | GNU/Linux Debian 3.1 Sarge (laptop 1)<br>GNU/Linux Backtrack 3 Beta (laptop 2) |
| *PROCESSORE* | Intel Pentium® M - Modello 725 (1.7 GHz, 400 FSB) con Hyper-Threading |
| *RAM* | Memoria DDR 333 SDRAM 512MB 333MHz Dual Channel Shared |
| *SCHEDA WIRELESS* | Intel® Pro Wireless 2200 802.11b/g |

*Tabella 2*

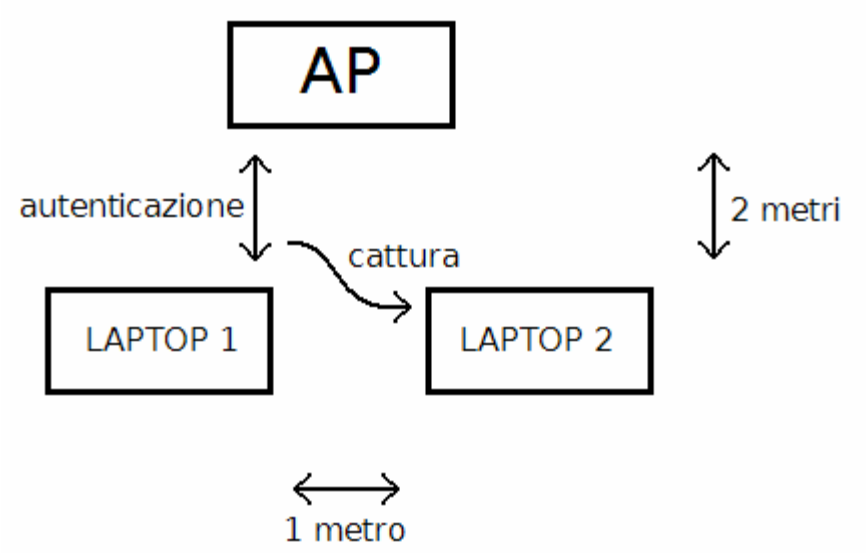


*Figura 1*

Infine, per attuare l'attacco a dizionario e considerata la necessità di avere a disposizione sufficiente potenza di calcolo, è stata utilizzata una macchina server con una configurazione minimale del sistema *GNU/Linux Debian 4.0*, con caratteristiche riportate in Tabella 3.

| | |
|---|---|
| *MODELLO* | Dell Poweredge SC430 |
| *OS* | GNU/Linux Debian 4.0r3 Etch |
| *PROCESSORE* | Intel Pentium® D – 3.0 GHz (dual core) |
| *RAM* | Memoria DDR2 1GB |

*Tabella 3*

Il *laptop* 1 è stato utilizzato per effettuare solo ed esclusivamente il processo di autenticazione con l'*AP*. Allo stesso tempo il *laptop* 2, sul quale era in esecuzione il sistema operativo *GNU/Linux Backtrack* ed il tool di cattura *airodump-ng* (componente del pacchetto *aircrack-ng*) è stato utilizzato per catturare i pacchetti di livello fisico scambiati durante l'autenticazione tra il *laptop* 1 e l'*AP*.

Per effettuare la cattura, dopo aver impostato il sistema in modalità *monitor*, è stato avviato il software airodump-ng con i seguenti parametri:

```
airodump-ng eth1 --channel 11 --bssid 00:11:22:33:44:55 -w filename
```

i cui parametri utilizzati sono elencati nella tabella sottostante:

| | |
|---|---|
| `eth1` | Interfaccia su cui catturare i pacchetti |
| `--channel 11` | Canale su cui limitare la cattura |
| `--bssid 00:11:22:33:44:55` | *BSSID* su cui limitare la cattura |
| `-w filename` | Nome del file in cui memorizzare i dati catturati |

Una volta avviato, il software airodump-ng si pone in modalità di cattura dei pacchetti, finchè non viene individuato un processo di *handshaking* tra due dispositivi. A questo punto memorizza tali pacchetti all'interno di un file (estensione *.cap*) per la successiva analisi e il processo di cattura viene terminato.

Nel caso quindi si conosca il momento preciso in cui una stazione sta effettuando l'handshaking, la fase di cattura risulta essere molto breve e verranno impiegate elevate risorse per l'analisi e l'elaborazione offline dei dati catturati.
Sono tuttavia reperibili in rete diversi tools (uno dei quali è *aireplay*, contenuto nella suite di aircrack) che forzano una particolare stazione alla disconnessione, per poi attendere la sua successiva riconnessione ed effettuare la cattura dell'*handshaking*. Non sono stati tuttavia utilizzati tali software, il cui impiego va aldilà del mero scopo didattico.

Come già anticipato, la tipologia di attacco qui adottata si riduce nella disponibilità di una potenza di calcolo sufficiente per raggiungere lo scopo (l'ottenimento della chiave condivisa) in un tempo ragionevole. Si è quindi optato per l'utilizzo di una macchina server che potesse adempiere a tale necessità e su di essa è stato eseguito il software aircrack-ng.

Dal momento che una ricerca esaustiva delle passphrase in modalità *brute-force* richiederebbe un tempo troppo elevato (v. paragrafo 4.3), si parte dall'assunto (in molti casi ragionevole) che le passphrase utilizzate siano appartenenti al linguaggio corrente e quindi si tenta un attacco mediante un particolare dizionario contenente una lista di parole candidate.

A tale scopo, è stato adottato un dizionario composto di 301'942 candidati, i cui termini sono stati ricavati da diversi dizionari messi a disposizione dalla University of Oxford al link **ftp://ftp.ox.ac.uk/pub/wordlists** .
Nello specifico, ipotizzando che la *passphrase* venga scelta da una persona di madrelingua inglese, sono stati utilizzati i seguenti dizionari, contenenti termini ritenuti con buona probabilità degli adeguati candidati per possibili *passphrase*:

- termini della lingua inglese
- acronimi
- nomi e cognomi di attori
- termini e nomi di personaggi pubblici, letterari, mitologici, biblici e cinematografici
- nomi di colleges
- passphrase comuni
- computer companies
- termini informatici
- nomi e cognomi comuni
- testi di canzoni famose
- nomi di cantanti e gruppi musicali
- termini sportivi
- nomi delle contee USA
- codici ZIP americani
- nomi di monumenti

Per avviare il processo di "cracking", è stato lanciato il software **aircrack-ng** con i seguenti parametri:

```
aircrack-ng filename.cap -w dictionaryfile
```

i cui parametri utilizzati sono elencati nella tabella sottostante:

| | |
|---|---|
| `filename.cap` | File di cattura che contiene l'*handshaking* |
| `-w dictionaryfile` | Nome del file contenente il dizionario da utilizzare |

I test effettuati hanno riguardato l'utilizzo di diverse parole chiave, per analizzare il diverso comportamento e la loro "resilienza" rispetto ad un tentativo di attacco a dizionario.

In particolare si è scelto di effettuare test con le seguenti caratteristiche:

1. passphrase di 8 lettere da dizionario inglese ("mobility")
2. passphrase di 8 lettere da dizionario inglese sostituendo lettera con numero ("mo8ility")
3. passphrase di 9 lettere da dizionario inglese ("messenger")
4. passphrase di 13 lettere da dizionario inglese ("compatibility")
5. passphrase di 8 lettere composta ("verycool")
6. passphrase di 13 lettere composta ("strangemonkey")
7. passphrase di 18 lettere composta ("thisisaverylongkey")
8. passphrase di 11 lettere costituita da nome e cognome ("chucknorris")
9. passphrase di 8 lettere e numeri casuali ("UX1B93HK")
10. passphrase di 13 lettere e numeri casuali ("ZF1A4JB7EY9OU")

La scelta di un attacco a dizionario piuttosto che un attacco a ricerca esaustiva è imposta dal tempo necessario per la "rottura" della protezione: come si vedrà nel capitolo relativo ai risultati (4.3), una ricerca esaustiva richiederebbe (in media) un tempo troppo elevato, rendendo vano ogni tentativo di rottura.

## 3. RICHIAMI TEORICI

Le prime implementazioni del protocollo 802.11 prevedevano sistemi di protezione non efficaci. Per superare le debolezze di questi sistemi è stata progettata l'estensione 802.11i, interamente dedicata al tema della sicurezza.
Sebbene lo standard 802.11i (noto anche come *WPA2*) sia implementato sulla maggioranza dei dispositivi in commercio e sia ad oggi largamente utilizzato, una buona parte delle installazioni attualmente in esercizio, sviluppate qualche anno fa, utilizza ancora i primi meccanismi di autenticazione e protezione resi disponibili.

I primi sistemi di autenticazione nati per le reti 802.11 sono di diversa natura e presentano diversi livelli di affidabilità:

- **Modalità chiusa basata su SSID**: la trasmissione non è criptata e i beacon non contengono l'*SSID* della

rete. Una stazione si può connettere solo se conosce l'*SSID*. Presenta un basso livello di sicurezza.

- **MAC Access Control List**: l'*AP* contiene una lista di indirizzi *MAC* ai quali è consentito l'accesso alla rete. Anche in questo caso la trasmissione non è criptata.
- **WEP**: la connessione è criptata con una chiave precedentemente condivisa e conosciuta da tutti i client. Per i dettagli sulla sicurezza di tale meccanismo si rimanda al paragrafo 3.1.
- **WPA**: meccanismo che migliora le debolezze del *WEP*. Ha alcune caratteristiche in comune con 802.11i, maggiormente noto come *WPA2*. E' il sistema di sicurezza maggiormente adottato nelle configurazioni di rete medio-piccole (residenziali o *SOHO*) ed è l'oggetto della presente sperimentazione.

I primi due sistemi di protezione sono facilmente aggirabili: per il primo, dato che durante la fase di autenticazione l'*SSID* è trasmesso in chiaro, è sufficiente catturare i pacchetti di autenticazione inviati da una stazione che conosce l'*SSID* e che si trova in quella fase. Per il secondo è possibile attuare il mascheramento del proprio indirizzo MAC (procedura nota come "*MAC spoofing*").

Anche il terzo sistema (*WEP*), per quanto maggiormente sicuro rispetto ai precedenti, è aggirabile e trova la sua debolezza nella chiave crittografica. Il sistema basato su *WPA* è più resistente rispetto al *WEP*, ma anch'esso aggirabile con particolari accorgimenti e sufficienti risorse a disposizione.

### 3.1 Il protocollo WEP

I componenti principali del WEP sono:

- una chiave ***K***, comune a tutte le stazioni che si collegano all'*AP*;
- l'uso di *RC4* come primitiva crittografica di tipo *stream cipher* con la chiave *K* e un vettore di inizializzazione (***IV***) di 24 bit;
- l'uso del *CRC-32* come protezione dell'integrità dei dati (***ICV***).

Lo schema a blocchi in figura 2 rappresenta il funzionamento del *WEP*.

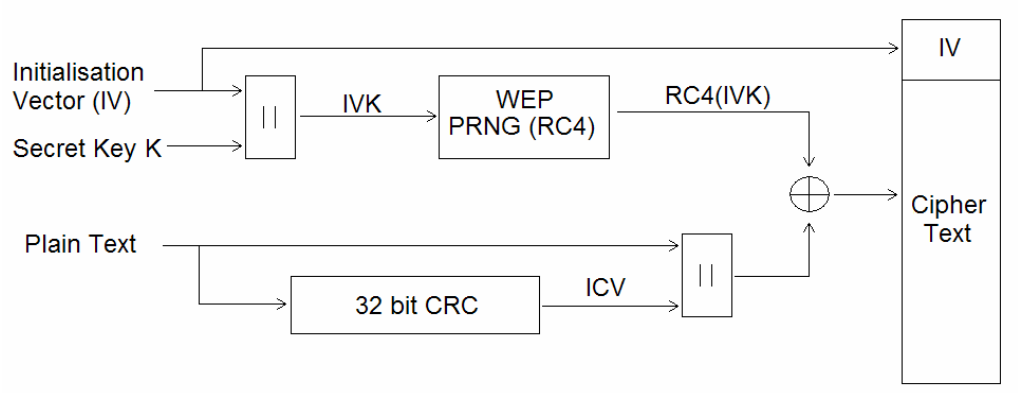


*Figura 2*

La chiave utilizzata è composta dalla concatenazione dell'*IV* e di *K*. Quest'ultima parte è comune per tutte le stazioni che si connettono all'*AP* ed ha una lunghezza che può variare da un minimo di 40 bit ad un massimo di 104 bit. Considerando che l'*IV* è formato da 24 bit, la chiave *IVK* risultante dalla combinazione di *K* e *IV* è compresa tra 64 e 128 bit. Su *IVK* viene eseguita la permutazione prevista dall'algoritmo *KSA (Key Scheduling Algorithm*) di RC4, dalla quale si genera uno stream di bit pseudocasuale, chiamato *keystream*. Successivamente, tramite l'operazione *XOR*, viene calcolato il messaggio criptato tra la concatenazione del testo in chiaro con l'*ICV* e il *keystream*. Assieme ad esso viene trasmesso in chiaro l'*IV*.

L'*ICV* è calcolato con un *CRC* (*cyclic redundancy check*) di 32 bit, il quale è usato per controllare che il testo decriptato sia uguale a quello trasmesso. Per fare questo il ricevitore calcola il *CRC* dopo aver decriptato il messaggio e lo confronta con l'*ICV* ricevuto.

Il meccanismo *WEP* prevede che ogni pacchetto sia criptato con una chiave diversa. Dato che la parte *K* della chiave è sempre uguale, si utilizza ogni volta un *IV* diverso. Questo è la causa della principale vulnerabilità del *WEP*, discussa nel prossimo paragrafo.

### 3.2 Vulnerabilità del protocollo WEP

Il sistema *WEP* presenta dei punti deboli derivanti dall'uso improprio delle primitive implementate. La funzione utilizzata per criptare i dati è una semplice *XOR*, efficace solamente nel caso in cui una chiave venga utilizzata una sola volta. Nel caso contrario, partendo da due messaggi criptati con la stessa chiave, è possibile ottenere l'*XOR* tra i due

messaggi e, laddove si conosca un messaggio, è possibile ottenere conseguentemente anche l'altro.

Il primo problema riguarda quindi la ripetitività della chiave, dovuta all'*IV*. Esso è l'unica sequenza di bit a differenziare due chiavi ed è formato da 24 bit, quantità che permette 16.777.216 differenti chiavi. Dato che il numero di frame trasmessi è elevato, è possibile riuscire a catturarne alcuni criptati con la stessa chiave.

Il secondo problema è legato al modo in cui l'*IV* viene ricavato. Lo standard non specifica come debba essere generato e, di conseguenza, i produttori di *NIC* sono liberi di calcolarlo in modi differenti. Quindi, considerando che l'*IV* è necessario al ricevitore per decriptare il messaggio ricevuto e che il ricevitore non conosce come il trasmettitore lo calcola, esso deve essere trasmesso in chiaro assieme al testo criptato. In questo modo solo una parte della chiave utilizzata è segreta.

Sono state inoltre scoperte alcune vulnerabilità nell'algoritmo *RC4* utilizzato nel *WEP*, le quali, unitamente alla conoscenza dei primi 3 byte della chiave, rendono veloce il calcolo della parte *K*. Il metodo inventato da S. Fluhrer, I. Mantin e A. Shamir rende veloce la scoperta di *K* anche nel caso in cui essa sia lunga 104 bit, dato che il tempo richiesto aumenta linearmente con l'aumentare della lunghezza della chiave. È stato dimostrato che esistono alcuni valori di *IV* che portano l'algoritmo *KSA* a rilasciare informazioni sulla chiave nei primi byte del *keystream*: catturando un numero sufficiente di pacchetti con questi *IV* è possibile ottenere la chiave crittografica utilizzata. Le informazioni sono ricavate utilizzando solo il primo byte del *keystream* prodotto dal vettore di stato di *KSA* in *RC4*.

Una terza vulnerabilità è fornita dal *CRC*: è infatti possibile modificarlo, senza conoscere la chiave di criptaggio, in maniera tale che il ricevitore non se ne accorga. In questa maniera si creano delle collisioni tra il *CRC* ricevuto e quello calcolato dal ricevitore.

### 3.3 Il protocollo WPA

Il protocollo di protezione *WPA* è stato creato nel tentativo di eliminare le debolezze del *WEP*, in attesa che fosse rilasciato il protocollo 802.11i (*WPA2*). Esistono due varianti di questo protocollo e la principale differenza risiede nella *PMK* (*Pairwise Master Key*), utilizzata da AP e stazioni per ricavare le chiavi temporanee (*PTK: Pairwise Transient Key*) da utilizzare durante la sessione dati. Esse sono stabilite durante il *4-way-handshake*, descritto nel paragrafo 3.4. Le due varianti sono:

- **Enterprise**: richiede un server IEEE 802.1x per l'autenticazione, il quale distribuisce ad ogni sessione una nuova *PMK* diversa per ogni stazione. È la soluzione che offre la sicurezza maggiore ma è anche la più costosa.
- **Personal**: utilizza una *passphrase* unica condivisa fra tutti gli utenti (*PSK*) come *PMK*. Essa non cambia (a meno di interventi dell'utente) e può essere lunga da 8 a 63 caratteri ASCII o 256 bit. Nel caso vengano utilizzati i caratteri ASCII, una *hash function* riduce i 504 bit della *passphrase* in 256 bit. È stata progettata per piccoli uffici o per uso personale, dato l'elevato costo di un server 802.1x per l'autenticazione.

*WPA* utilizza la primitiva crittografica *RC4*, usata anche nel *WEP*, con una chiave migliore. Essa è lunga 128 bit ed è creata in 2 fasi successive dal protocollo *TKIP* (*Temporal Key Integrity Protocol*). Il vantaggio dell'uso di *RC4* è dato dalla semplicità con cui i sistemi basati su *WEP* possono essere aggiornati al WPA (è necessario solo un aggiornamento del firmware).

Nella prima fase vengono combinate assieme tre differenti sequenze di bit:

- ***Base Key***: chiave condivisa con l'*AP* creata durante la fase di autenticazione dela stazione (PTK). È diversa ad ogni nuova autenticazione.
- ***TA*** (*Transmitter Address*): chiave lunga 48 bit che contiene il *MAC* della stazione trasmittente. Questo campo sarà quindi diverso in base alla direzione della trasmissione.
- ***IV***: è l'*initialization vector* che è utilizzato anche nel *WEP*. In questo caso è lungo 48 bit ed è incrementato ad ogni nuovo pacchetto.

Nella seconda fase viene combinato il risultato della prima fase con l'*IV*, creando una chiave, diversa per ogni pacchetto, che verrà utilizzata dalla primitiva crittografica *WEP*. Lo schema di quanto è stato descritto è rappresentato in figura 3.

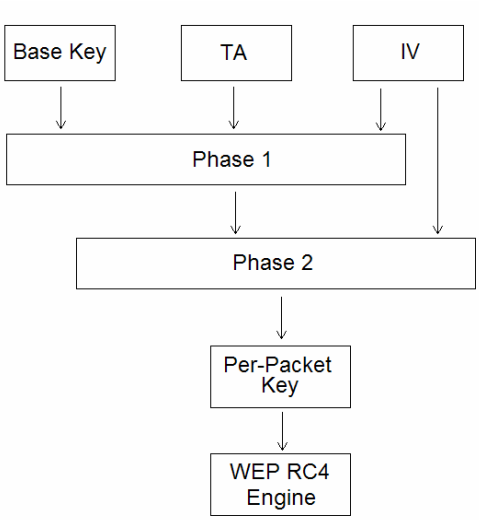


*Figura 3*

Con l'utilizzo del *TKIP* si eliminano tutti i problemi che si hanno con il *WEP*, in particolare viene migliorato l'uso dell'*IV* in quanto:

- è composto da 48 bit per evitare la cattura di due pacchetti con lo stesso *IV*;
- deve essere calcolato in maniera incrementale per evitare che debba essere trasmesso in chiaro (la *NIC* che riceve il pacchetto sa come ricavare il suo valore). Calcolando l'IV in questo modo si ottiene anche un secondo vantaggio: la trasmissione è immune ai *Replay Attack*; infatti, nel caso arrivassero alla *NIC* dei frame con un *IV* già utilizzato, essi vengono scartati.

Per quanto riguarda il controllo di integrità dei dati, il *WPA* sostituisce il *CRC* del *WEP* con un campo *MIC*, ottenuto dall'algoritmo chiamato ***Michael***. Composto da 8 byte e criptato assieme ai dati, è progettato per resistere agli attacchi cui è vulnerabile il *CRC* del *WEP*.

### 3.4 Vulnerabilità del protocollo WPA

Anche il *WPA* non è esente da vulnerabilità. La sua debolezza sta nel fatto che la *Base Key* è concordata dall'*AP* e dalla stazione durante la fase di autenticazione, composta da quattro messaggi. Lo scambio dei messaggi e le operazioni eseguite dalla stazione e dall'AP sono mostrate in figura 4.

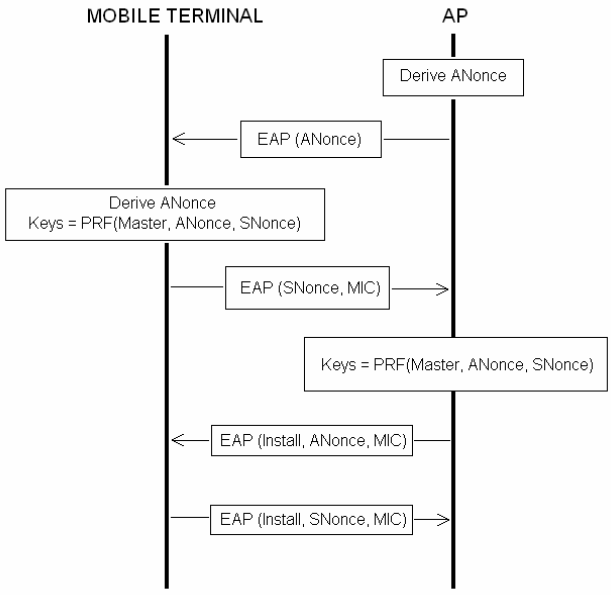


*Figura 4*

Nella figura 4 si distinguono due particolari funzioni:

- ***PRF***: *Pseudo Random Function*, serve per generare i bit pseudo-casuali;
- ***EAP***: *Extensible Authentication Protocol*, è un protocollo generale per l'autenticazione *Point-to-Point* in grado di supportare meccanismi di autenticazione multipli.

Viene inoltre mostrata nel dettaglio la sequenza dei quattro messaggi di *handshake*, spiegati nelle righe successive:

- **AP**: calcola un valore random chiamato *ANonce* e lo manda alla stazione (mobile terminal) tramite il protocollo *EAP*;
- **Stazione**: calcola un valore random (*SNonce*) e la *PTK* utilizzando l'*ANonce* e lo *SNonce*. Successivamente manda un messaggio all'*AP* con lo *SNonce* e un codice per controllare l'integrità del messaggio (*MIC*), criptato con la chiave appena calcolata.
- **AP**: calcola il *PTK* come fatto precedentemente dal terminale. Manda un messaggio con *GTK* (*Groupwise Transient Key*), la quale è usata per decriptare i messaggi *broadcast* e *multicast*, con un altro *MIC*.
- **Stazione**: conferma all'*AP* la ricezione del messaggio.

Nel caso di *WPA-enterprise* ogni sessione utilizza una chiave ricavata dalla *PMK*. La *PMK* viene sostituita ad ogni nuova procedura di autenticazione ed è fornita dal server utilizzato per l'autenticazione. Anche nel caso di *WPA-personal* la chiave utilizzata durante la sessione dati è temporanea ed è ricavata da una *PMK* durante l'autenticazione, tuttavia la *PMK* è proprio la *passphrase PSK*, la quale non varia da una sessione all'altra.

La chiave temporanea è calcolata basandosi su:

- *PMK*
- *SNonce*
- *ANonce*
- indirizzi *MAC* della stazione dell'*AP*

La *PMK* è distribuita dal server di autenticazione IEEE 802.1x nel caso di *WPA-enterprise* o è già installata sull'*AP* e sulle stazioni nella configurazione *WPA-personal*.
Lo *SNonce* e l'*ANonce* sono necessari alla stazione e all'AP per poter calcolare la *PTK* e, dato che sono generati da due entità separate, devono essere inviati l'uno all'altro. La loro trasmissione, come quella dell'indirizzo *MAC*, avviene in chiaro, dato che ancora nessuna chiave è stata impostata.

Catturando i pacchetti che contengono questi due campi (trasmessi nel *4-way-handshake*) è possibile costruire un attacco *brute force* nel quale si provano tutte le combinazioni alfa-numeriche di *PMK*. Per ogni possibile chiave ricavata si calcola il *MIC* che andrebbe inviato nel secondo pacchetto e lo si confronta con quello catturato. Nel caso in cui essi coincidano la chiave trovata è la *PMK* utilizzata nella sessione. Tuttavia non è possibile provare tutte le combinazioni possibili, poiché sarebbero necessarie risorse (tempo e capacità di calcolo) troppo elevate (v. paragrafo 4.3).

L'unica maniera per velocizzare la ricerca della chiave è l'utilizzo di un dizionario, data la maggiore probabilità che la chiave sia una parola di senso compiuto. L'utilizzo di combinazioni casuali di lettere e numeri rende però anche gli attacchi a dizionario sostanzialmente inutili.

L'attacco *brute-force* ha senso solo nel caso di *WPA-personal*, dove la *PMK* coincide con la *PSK*, perchè in questo caso la *PSK* non varia da una sessione all'altra. Nel caso di *WPA-enterprise* l'attacco *brute-force* è inutile, dato che ad ogni nuova sessione viene utilizzata una *PMK* diversa, rendendo inutilizzabile l'eventuale *PMK* precedentemente calcolata con l'attacco.

## 4. RISULTATI SPERIMENTALI

### 4.1 Analisi dei risultati

Vengono di seguito riportati i risultati sperimentali ottenuti. Come anticipato, sono state adottate 10 diverse tipologie di *passphrase*, per analizzarne la resistenza rispetto ad un attacco a dizionario.

Per ogni test viene riportata la *passphrase* originale, l'eventuale successo del tentativo di "rottura" e il relativo tempo di decodifica impiegato. Come ampiamente sottolineato, l'immunità all'attacco della parola oggetto dell'esperimento non implica l'immunità assoluta, ma solo ed esclusivamente l'immunità all'attacco basato sul dizionario della presente sperimentazione.

**TEST 1:** passphrase di 8 lettere da dizionario inglese

```
[00:07:28] 68144 keys tested (150.38 k/s)

KEY FOUND! [ mobility ]

Master Key     : 46 2D E3 9B 67 18 18 A8 30 9F FA 3E 90 CA 78 F2
                 D5 1F FF D3 B2 34 AE 89 01 73 0C 3D 20 A7 27 7D

Transcient Key : B3 A1 6B 65 FF DE 66 48 A0 9D E4 C9 3D 4D A1 66
                 82 7F A6 42 C5 25 C9 83 F8 F4 28 DB 1A B1 3A 16
                 27 AC A3 D8 E6 34 27 88 2A 4E 66 BD 5A 78 BA F0
                 E8 55 A0 20 62 A2 81 89 DE 33 2A D3 D5 C2 28 6F

EAPOL HMAC     : F6 D3 6F 24 DC 32 FC C0 DB 53 07 9F 06 2C F1 A0
```

| **PASSPHRASE** |
|---|
| mobility |
| **PASSPHRASE INDIVIDUATA** |
| SI |
| **TEMPO IMPIEGATO** |
| 00:07:28 |
| **CHIAVI TESTATE** |
| 68144 |
| **VELOCITÀ MEDIA** |
| 150.38 chiavi/s |

Il primo test mostra come una chiave appartenente al linguaggio corrente e al dizionario della lingua inglese sia ottenibile in pochi minuti con un semplice dizionario e con ridotte capacità di calcolo. Per tale motivo una parola di questo genere non dovrebbe essere mai adottata come chiave *WPA*.

E' poi importante sottolineare come l'utilizzo di questa tipologia di chiave possa permettere di trarre il vantaggio massimo dall'utilizzo di un sistema a dizionario rispetto ad un sistema a ricerca esaustiva: la tipologia di *passphrase* adottata è esattamente quella per cui sono stati creati i sistemi a dizionario, ovvero una *passphrase* semplice e comune.

**TEST 2:** passphrase di 8 lettere da dizionario inglese sostituendo lettera con numero

```
[00:33:06] 301942 keys tested (150.41 k/s)

Passphrase not in dictionary
```

| **PASSPHRASE** |
|---|
| mo8ility |
| **PASSPHRASE INDIVIDUATA** |
| NO |
| **TEMPO IMPIEGATO** |
| 00:33:07 |
| **CHIAVI TESTATE** |
| Intero dizionario |
| **VELOCITÀ MEDIA** |
| 150.41 chiavi/s |

Il test 2 mostra come, in presenza di un minimo cambiamento della *passphrase* (la cifra "8" al posto della lettera "b"), il sistema di cracking non sia più in grado di rilevare la *passphrase*, poiché non contenuta nel dizionario.
Ciò non significa che la *passphrase* a questo punto sia sicura: potrebbe ad esempio essere adottata una variante dell'attacco a dizionario, che, oltre alla verifica di ogni singola parola del dizionario, tenta il "matching" anche con alcune varianti comuni della parola stessa.

E' prassi comune infatti, nella scelta di una *passphrase*, sostituire alcune lettere con alcuni numeri secondo una associazione prefissata: spesso la lettera "b" viene sostituita con un 8, la lettera "l" con un 1, la lettera "t" con un 7, la lettera "o" con uno 0, la lettera "s" con un 5 e così via. Un eventuale algoritmo alternativo potrebbe sfruttare questo schema per analizzare anche possibili varianti delle parole del dizionario, aumentando così il tempo di analisi, ma anche la probabilità di rilevare una possibile *passphrase* codificata secondo tale schema.

**TEST 3:** passphrase di 9 lettere da dizionario inglese

```
[00:07:18] 66624 keys tested (149.46 k/s)

KEY FOUND! [ messenger ]

Master Key     : F5 F8 F4 9C 28 70 3E C8 E8 B5 F3 72 C2 5D 64 8E
                 24 FB 98 3B 2B A5 E9 8E E1 E1 03 A6 48 D7 34 DD

Transcient Key : EF C6 98 27 E9 B1 CB AC 1F B0 8E 37 5B EE 12 94
                 74 CC 74 B3 CD 68 F4 F0 19 88 DC 49 AC D6 46 DE
                 03 EB 12 EE 85 1C 74 D2 FE 2E 29 94 89 52 0D A9
                 67 13 92 50 17 19 F8 F6 9F 72 A2 89 F1 7A 34 28

EAPOL HMAC     : 1F 83 09 39 48 B3 62 DC 88 B9 54 90 CA AD D2 3C
```

| **PASSPHRASE** |
|---|
| messenger |
| **PASSPHRASE INDIVIDUATA** |
| SI |
| **TEMPO IMPIEGATO** |
| 00:07:18 |
| **CHIAVI TESTATE** |
| 66624 |
| **VELOCITÀ MEDIA** |
| 149.46 chiavi/s |

Il test 3 mostra come in un attacco a dizionario la lunghezza della *passphrase* sia ininfluente: pur avendo aumentato la lunghezza della parola di una singola lettera il tempo impiegato è pressoché simile a quello del test 1. Ciò conferma che il tempo impiegato non è funzione della lunghezza, quanto più della posizione della *passphrase* (se presente) all'interno del dizionario stesso.

**TEST 4:** passphrase di 13 lettere da dizionario inglese

```
[00:02:29] 22602 keys tested (149.65 k/s)


KEY FOUND! [ compatibility ]


Master Key     : E2 AF 64 BB 44 BE 6A 2C F9 47 AE 60 D9 CC FF 12
                 1F 3E 0E F3 14 A6 61 AA 10 5A B6 29 F0 EE 4D D1

Transcient Key : 63 8F 64 81 E1 54 BE 8C ED 85 8A 02 04 62 91 14
                 34 6E E5 18 21 AB F9 3A 1F EF A6 F0 94 3C 49 5B
                 C2 B8 2B F2 A9 FB A9 1B 75 16 76 13 15 C8 47 41
                 52 4D F6 34 A6 C9 8A 37 56 BD 5A 38 A2 4E 12 48

EAPOL HMAC     : E8 6E 1C 1E E3 37 E8 8A BA 2C C5 6C D8 66 3D 28
```

| **PASSPHRASE** |
|---|
| compatibility |
| **PASSPHRASE INDIVIDUATA** |
| SI |
| **TEMPO IMPIEGATO** |
| 00:02:29 |
| **CHIAVI TESTATE** |
| 22602 |
| **VELOCITÀ MEDIA** |
| 149.65 chiavi/s |

Il test 4 conferma le conclusioni tratte dal test 3: in questo caso la parola è ben più lunga delle precedenti (13 lettere anziché 9 oppure 8) e viene decodificata in un tempo decisamente minore. Ciò è dovuto al fatto che il dizionario, prima di essere utilizzato, è stato ordinato alfabeticamente ed essendo la *passphrase* del test 4 antecedente rispetto alle posizioni dei test 1 e 3, viene decodificata in un tempo minore.

**TEST 5:** passphrase di 8 lettere composta

```
[00:33:06] 301942 keys tested (151.25 k/s)

Passphrase not in dictionary
```

| **PASSPHRASE** |
|---|
| verycool |
| **PASSPHRASE INDIVIDUATA** |
| NO |
| **TEMPO IMPIEGATO** |
| 00:33:06 |
| **CHIAVI TESTATE** |
| Intero dizionario |
| **VELOCITÀ MEDIA** |
| 151.25 chiavi/s |

I test 5, 6 e 7 hanno lo scopo di valutare quanto sia resistente una *passphrase* composta da diverse parole del dizionario. Il test 5 utilizza due parole distinte ma che nella lingua inglese spesso ricorrono assieme. I risultati mostrano che il tentativo di "rottura" non ha avuto successo, dal momento che la *passphrase* non era contenuta all'interno del dizionario.

**TEST 6:** passphrase di 13 lettere composta

```
[00:33:07] 301942 keys tested (150.49 k/s)

Passphrase not in dictionary
```

| **PASSPHRASE** |
|---|
| strangemonkey |
| **PASSPHRASE INDIVIDUATA** |
| NO |
| **TEMPO IMPIEGATO** |
| 00:33:07 |
| **CHIAVI TESTATE** |
| Intero dizionario |
| **VELOCITÀ MEDIA** |
| 150.49 chiavi/s |

Il test 6 utilizza l'accostamento di due differenti parole, la cui ricorrenza congiunta all'interno del linguaggio corrente però non è così frequente come quella del test 5.

Anche qui il tentativo di "rottura" fallisce, tuttavia è ragionevole supporre che, se le parole del test 6 sono meno frequenti rispetto a quelle del test 5, la *passphrase* del test 6 possa essere considerata più sicura rispetto alla precedente, restando comunque entrambe delle *passphrase* decisamente deboli.
Sarebbe infatti bastata una variante dell'attacco a dizionario (anziché testare solo le parole del dizionario, tentare anche diverse combinazioni tra esse) per rilevare entrambe le *passphrase*. E' palese come una variante di attacco di questo tipo possa far lievitare esponenzialmente i tempi di decodifica, ma aumenta di conseguenza anche il dominio delle possibili *passphrase* rilevabili.

Per far fronte all'elevato tempo di decodifica, come evidenziato successivamente, è comunque possibile parallelizzare l'algoritmo di decodifica attraverso un *computer cluster*.

**TEST 7:** passphrase di 18 lettere composta

[00:33:06] 301942 keys tested (151.21 k/s)

Passphrase not in dictionary

| |
|---|
| **PASSPHRASE** |
| thisisaverylongkey |
| **PASSPHRASE INDIVIDUATA** |
| NO |
| **TEMPO IMPIEGATO** |
| 00:33:06 |
| **CHIAVI TESTATE** |
| Intero dizionario |
| **VELOCITÀ MEDIA** |
| 151.21 chiavi/s |

Il test 7 utilizza come *passphrase* una intera frase. La chiave è resistita all'attacco, dal momento che il dizionario conteneva singole parole e non combinazioni tra esse.

**TEST 8:** passphrase di 11 lettere costituita da nome e cognome

[00:33:06] 301942 keys tested (150.30 k/s)

Passphrase not in dictionary

| |
|---|
| **PASSPHRASE** |
| chucknorris |
| **PASSPHRASE INDIVIDUATA** |
| NO |
| **TEMPO IMPIEGATO** |
| 00:33:06 |
| **CHIAVI TESTATE** |
| Intero dizionario |
| **VELOCITÀ MEDIA** |
| 150.30 chiavi/s |

Il test 8 ha lo scopo di valutare se una *passphrase* formata dal nome e cognome di un personaggio famoso possa essere ricavata in tempi brevi, per testare quindi l'utilità di uno dei dizionari (nomi e cognomi di attori) utilizzati per creare il dizionario composto utilizzato in questo esperimento.
Il tentativo di decodifica, come è possibile notare, è fallito. Dopo una attenta analisi del dizionario, è stato infatti rilevato che esso conteneva sia il nome che il cognome, ma utilizzandoli come termini separati. Non tentando quindi la combinazione tra di essi, la *passphrase* non è stata ricavata.
Tale passphrase è comunque da ritenersi debole, in quanto sarebbe bastato un semplice dizionario che contenesse i nomi e i cognomi di attori come termini singoli perché la *passphrase* oggetto del test 8 potesse essere ricavata.

**TEST 9:** passphrase di 8 lettere e numeri casuali

[00:33:06] 301942 keys tested (150.71 k/s)

Passphrase not in dictionary

| |
|---|
| **PASSPHRASE** |
| UX1B93HK |
| **PASSPHRASE INDIVIDUATA** |
| NO |
| **TEMPO IMPIEGATO** |
| 00:33:06 |
| **CHIAVI TESTATE** |
| Intero dizionario |
| **VELOCITÀ MEDIA** |
| 150.71 chiavi/s |

Gli ultimi due test (9 e 10) mostrano il tentativo di rottura di una *passphrase* con lettere e numeri casuali. Il test 9, in particolare, analizza una *passphrase* casuale di lunghezza 9, che non viene ricavata.
Tale tipologia di *passphrase*, infatti, è da ritenersi la più sicura rispetto ad un attacco a dizionario, in quanto gode delle proprietà di **complessità** ed **unicità** che rendono questo schema di attacco molto debole.

**TEST 10:** passphrase di 13 lettere e numeri casuali

[00:33:07] 301942 keys tested (151.17 k/s)

Passphrase not in dictionary

| |
|---|
| **PASSPHRASE** |
| ZF1A4JB7EY9OU |
| **PASSPHRASE INDIVIDUATA** |
| NO |
| **TEMPO IMPIEGATO** |
| 00:33:07 |
| **CHIAVI TESTATE** |
| Intero dizionario |
| **VELOCITÀ MEDIA** |
| 151.17 chiavi/s |

Il test 10, a differenza del precedente, analizza una *passphrase* di lunghezza 13, di lettere e numeri casuali. Come nel test 9, il tentativo di decodifica fallisce.

Anche in questo caso la *passphrase* è ragionevolmente sicura rispetto ad un attacco a dizionario. Tuttavia è bene considerare anche l'eventuale resilienza rispetto ad un attacco a ricerca esaustiva, descritto successivamente nel paragrafo 4.3. Con tale attacco, infatti, i tempi di decodifica aumentano esponenzialmente con la lunghezza della *passphrase* e per ottenere un maggiore livello di sicurezza, è consigliabile adottare *passphrase* di elevata lunghezza.

**4.2 Riepilogo**

| PASSPHRASE | INDIVIDUATA | TEMPO DECOD. | CHIAVI TESTATE | VELOCITÀ MEDIA |
|---|---|---|---|---|
| mobility | SI | 00:07:28 | 68'144 | 150.38 chiavi/s |
| mo8ility | NO | - | TUTTE (301'942) | 150.41 chiavi/s |
| messenger | SI | 00:07:18 | 66'624 | 149.46 chiavi/s |
| compatibility | SI | 00:02:29 | 22'602 | 149.65 chiavi/s |
| verycool | NO | - | TUTTE (301'942) | 151.25 chiavi/s |
| strangemonkey | NO | - | TUTTE (301'942) | 150.49 chiavi/s |
| thisisaverylongkey | NO | - | TUTTE (301'942) | 151.21 chiavi/s |
| chucknorris | NO | - | TUTTE (301'942) | 150.30 chiavi/s |
| UX1B93HK | NO | - | TUTTE (301'942) | 150.71 chiavi/s |
| ZF1A4JB7EY9OU | NO | - | TUTTE (301'942) | 151.17 chiavi/s |

*Tabella 4*

Come è possibile notare dalla tabella 4, le uniche *passphrase* che sono state rilevate sono solo ed esclusivamente parole semplici e comuni, contenute all'interno del dizionario della lingua inglese.
Tutte le altre *passphrase* (varianti, parole composte, casuali) si sono rivelate resistenti all'attacco basato sul dizionario oggetto di questa analisi.

**4.3 Vantaggi e svantaggi di una ricerca esaustiva**

Il processo di *cracking* sulla macchina indicata in tabella 3 è avvenuto alla velocità di circa **150 chiavi/secondo**. Ciò significa che – considerando una possibile *passphrase* **fino ad 8 caratteri**, composta solo da lettere e numeri – un attacco *brute-force* in modalità esaustiva avrebbe impiegato nel peggiore dei casi un tempo pari a:

$$t = \frac{(26+10)^8}{150} \cong 1.88 \cdot 10^{10} s \cong 596 anni$$

Pur essendo quindi la ricerca esaustiva il metodo di "rottura" ottimo (esso garantisce in ogni caso l'ottenimento della *passphrase*), è chiara la necessità di restringere la ricerca con un attacco a dizionario. Si tenga presente che in realtà la lunghezza di una *passphrase WPA* può arrivare **fino a 63 caratteri** e accetta **qualsiasi carattere *printable ASCII***!

E' inoltre palese che effettuare un attacco in modalità esaustiva richiederebbe elevatissime capacità di calcolo e molto tempo, da cui si deduce che utilizzando *passphrase* di sufficiente lunghezza il protocollo *WPA* è ragionevolmente sicuro rispetto agli attacchi esaustivi, considerato che la complessità cresce **esponenzialmente** con la lunghezza della *passphrase*.

Infatti, utilizzando una *passphrase* di 13 caratteri, il tempo per una ricerca esaustiva sale a:

$$t = \frac{(26+10)^{13}}{150} \cong 1.14 \cdot 10^{18} s \cong 3.61 \cdot 10^{10} anni$$

Per una *passphrase* di lunghezza 20, sempre composta da lettere e numeri, il tempo per la "rottura" sale a:

$$t = \frac{(26+10)^{20}}{150} \cong 8.91 \cdot 10^{28} s \cong 2.83 \cdot 10^{21} anni$$

Si nota quindi che, pur mantenendo fisso il vincolo di utilizzo di sole lettere e numeri, una *passphrase* di 20 caratteri garantisce un ragionevole livello di protezione contro l'attacco esaustivo.

Un attacco totalmente esaustivo richiederebbe l'analisi di tutte le possibili combinazioni che formano una chiave a 256 bit, impiegando pertanto un tempo pari a:

$$t = \frac{2^{256}}{150} \cong 7.72 \cdot 10^{74} s \cong 2.45 \cdot 10^{67} anni$$

Risulta importante sottolineare infine come un eventuale **algoritmo di ricerca esaustiva** sia totalmente parallelizzabile e quindi il tempo di decodifica può essere ridotto **linearmente** mediante la suddivisione dell'operazione di ricerca con l'utilizzo di un cluster di calcolatori.

In un **attacco a dizionario**, invece, il tempo impiegato per la decodifica è dipendente dalle capacità di calcolo, ma anche e soprattutto dalla presenza e dall'eventuale posizione all'interno del dizionario della *passphrase WPA*.

A differenza di quanto avviene con un attacco a ricerca esaustiva, in un attacco a dizionario la lunghezza della *passphrase* non è sufficiente a garantire un ragionevole livello di sicurezza. *Passphrase* di elevata lunghezza ma contenute all'interno del dizionario di attacco possono essere ottenute con semplicità. Per contro, *passphrase* brevi, ma complesse, possono resistere ad un attacco a dizionario.

Ciò che in realtà rende immune una *passphrase* da un attacco a dizionario sono la **complessità** e l'**unicità** della stessa: più si avvicina ad una combinazione di lettere e numeri casuali, più sarà elevata la possibilità che NON sia contenuta all'interno del dizionario di attacco.

Anche in questo caso, comunque, l'operazione di ricerca è totalmente parallelizzabile.

## 5. CONCLUSIONI

La presente sperimentazione aveva lo scopo di verificare il livello di affidabilità del protocollo di protezione *WPA* adottato all'interno di reti wireless.

I risultati ottenuti hanno mostrato come il concetto di "sicurezza" sia in realtà relativo e strettamente legato al tipo di attacco attuato per "rompere" il protocollo di protezione ed entrare in possesso della *passphrase*.

Il tipo di schema d'attacco adottato è uno schema a dizionario, che ha mostrato le seguenti caratteristiche:

- Funzionamento ottimo in caso di *passphrase* semplici, comuni e appartenenti al linguaggio corrente
- Brevi tempi di calcolo
- Inefficacia in caso di *passphrase* complesse

In realtà le prestazioni di questo tipo di attacco dipendono fortemente dal tipo di dizionario che viene utilizzato e da quanto esso copra tutte le possibili diverse varianti delle parole della lingua corrente. Inoltre il livello di sicurezza non dipende dalla lunghezza della *passphrase*, ma dalla sua complessità.

L'attacco a dizionario e l'attacco esaustivo sono gli estremi di un intero insieme di attacchi possibili: per aumentare l'efficacia sarebbe infatti possibile basarsi su un dizionario e dai termini appartenenti ad esso analizzare possibili variazioni o combinazioni. E' evidente che il tempo di analisi aumenterebbe esponenzialmente (problema comunque risolvibile grazie alla forte parallellizzazione permessa dall'algoritmo che ridurrebbe, seppure in modo lineare, i tempi di calcolo), ma si otterrebbe una tipologia di attacco decisamente più efficace.

E' pertanto possibile affermare che il protocollo *WPA* è ragionevolmente sicuro rispetto all'attacco *brute-force* in presenza di una **passphrase complessa** (per l'attacco a dizionario) e di **elevata lunghezza** (per l'attacco esaustivo).

In sede di scelta della *passphrase* è comunque consigliabile tenere ben presente gli aspetti evidenziati in questa sperimentazione ed effettuare la scelta in un ottica di "possibili attacchi attuabili", e quindi sul fatto che la *passphrase* possa essere contenuta in un eventuale dizionario di attacco con la minima probabilità possibile.