Aspetti di sicurezza dei protocolli per wireless (WEP / WPA / TKIP)

Alessio Caprari <acaprari@cs.unibo.it>
Francesco Iezzi <iezzi@cs.unibo.it>
Gian Renato Maffini <maffini@cs.unibo.it>

1 ottobre 2004

Indice

1 Le reti wireless

1.1 La situazione attuale

La tecnologia wireless per le reti locali (WLAN) è uno strumento in veloce espansione. Grazie al progressivo abbassamento dei prezzi ed al raggiungimento di velocità paragonabili a quelle delle reti cablate, molte aziende ed amministratori di rete stanno decidendo di passare alla tecnologia senza fili.

Negli ultimi anni, il protocollo di riferimento, quando si parla di reti wireless, è lo standard IEEE 802.11 [1]. La più importante applicazione dei prodotti che lo implementano è all'interno delle reti locali (LAN). In queste reti il traffico è principalmente di tipo TCP/IP e non presenta grosse differenze rispetto a quello presente nelle comuni reti cablate [2].

Nelle case o nei piccoli uffici, la tecnologia WLAN è usata principalmente per fornire accesso ad Internet, ad esempio attraverso connessioni di tipo xDSL. Nell'ambito domestico, quando si ha a disposizione una connessione wireless, spesso, non si ha nessuna rete cablata tradizionale.

Nelle aziende, invece, solitamente si riscontra una situazione mista: la rete cablata, che costituisce l'infrastruttura principale, a volte viene ampliata da collegamenti wireless che forniscono connettività in particolari punti, come ad esempio le sale conferenza, mentre in altri casi il wireless è utilizzato come mezzo di comunicazione per qualsiasi host utilizzato da un utente.

Il protocollo 802.11, inoltre, per mezzo della modalità di connessione denominata *ad-hoc*, consente agli utenti di creare gruppi di connettività senza la necessità di alcuna infrastruttura.

1.2 Funzionamento di una rete 802.11

Una rete WLAN di tipo IEEE 802.11 è costituita da un gruppo di stazioni (i nodi della rete wireless) collocate all'interno di una rete fisica delimitata, dove ogni singola stazione è in grado di effettuare una comunicazione radio con una stazione base. Esistono due tipi diversi di configurazioni possibili per le WLAN [3]:

Ad-hoc in questa modalità non si è in grado di comunicare con un network esterno senza l'aiuto di protocolli di routing addizionali. È normalmente creata per permettere a più stazioni wireless di comunicare direttamente tra loro, con requisiti minimi in termini di hardware e gestione.

Infrastructure-based questa configurazione è composta da uno o più Basic Service Set (BSS). Ogni stazione ha esattamente un link BSS per connettersi all'infrastruttura, il Distribuition System (DS), che permette l'accesso a network esterni. Il nodo che collega una stazione al DS, chiamato Access Point, commuta i pacchetti, di una stazione contenuta nel BSS, verso il DS.

1.3 Problemi di sicurezza delle WLAN

Una qualsiasi trasmissione di rete, sia essa effettuata con connessioni cablate o senza fili, presenta da sempre diverse problematiche relative alla sicurezza [4]. Ad esempio, in una LAN, essendo consentito accesso multiplo al canale di trasmissione, qualsiasi host che sia connesso alla rete ha la possibilità di intercettare una qualunque trasmissione effettuata all'interno della stessa.

Le reti wireless però, trasmettendo dati per mezzo delle onde radio, presentano anche altri problemi dipendenti dalle caratteristiche del canale di comunicazione utilizzato. Quando una trasmissione avviene attraverso l'aria, l'intercettazione e la manipolazione dei dati diventano operazioni banali da parte di chiunque possieda un'adeguata apparecchiatura; si rende così necessario lo sviluppo di meccanismi aggiuntivi per la protezione delle comunicazioni [5].

In particolare i problemi principali che riguardano una WLAN possono essere suddivisi in queste categorie:

Riservatezza è necessario impedire che si possano intercettare i dati trasmessi attraverso il canale.

Access Control l'accesso alla rete deve essere consentito solamente agli host autorizzati.

Integrità dei dati si deve evitare che si possano manomettere i messaggi trasmessi.

2 Wired Equivalent Privacy (WEP)

Lo standard IEEE 802.11 definisce un meccanismo per la riservatezza dei dati conosciuto come Wired Equivalent Privacy (WEP) [1], il quale si pone l'obiettivo di raggiungere un livello di sicurezza pari a quello delle reti cablate.

2.1 La crittografia dei dati

Il processo di modifica dei dati, con lo scopo di nasconderne le informazioni contenute, è chiamato "crittografia" (*encryption*). I dati che non sono crittografati sono chiamati "testo in chiaro" (*plaintext*), nelle formule indicati con la lettera P, mentre i dati cifrati *(cyphertext)* sono indicati con C. Il processo inverso che permette di trasformare i dati cifrati in testo in chiaro è detto "decrittografare" (*decryption*).

L'algoritmo di crittografia è una funzione matematica usata per cifrare o decifrare i dati. I moderni algoritmi di crittografia utilizzano una chiave, indicata con k, per modificare il loro output. La funzione di cifratura E opera su P per produrre il messaggio cifrato C:

$$E_k(P) = C$$

Nel processo inverso, invece, la funzione di decifratura D opera su C per produrre il testo in chiaro P:

$$D_k(C) = P$$

Si può notare che questo processo di cifratura e decifratura è simmetrico, infatti la stessa chiave può essere usata in entrambe le operazioni:

$$D_k(E_k(P)) = P$$

2.2 L'algoritmo alla base del WEP

L'algoritmo del WEP è una sequenza di operazioni che permette di modificare un blocco di testo in chiaro calcolandone lo XOR bit a bit con una chiave pseudocasuale di uguale lunghezza. Questa sequenza pseudocasuale è generata dall'algoritmo stesso, come mostrato nella figura 1 che illustra lo schema di funzionamento del WEP. In essa si può notare che tutto il processo di codifica prevede la presenza di una *chiave segreta* che costituisce uno degli input più importanti dell'algoritmo.

Tutti gli host appartenenti ad un'unica rete wireless, che vogliano comunicare tra loro, devono possedere la stessa chiave segreta. Le modalità con cui questa chiave debba essere scelta e si debba diffondere tra i vari host, non sono specificate dal WEP e sono quindi affidate ad un sistema esterno. Tipicamente è l'amministratore stesso di una rete wireless che si preoccupa di scegliere la chiave e di comunicarla successivamente a tutti gli utenti.

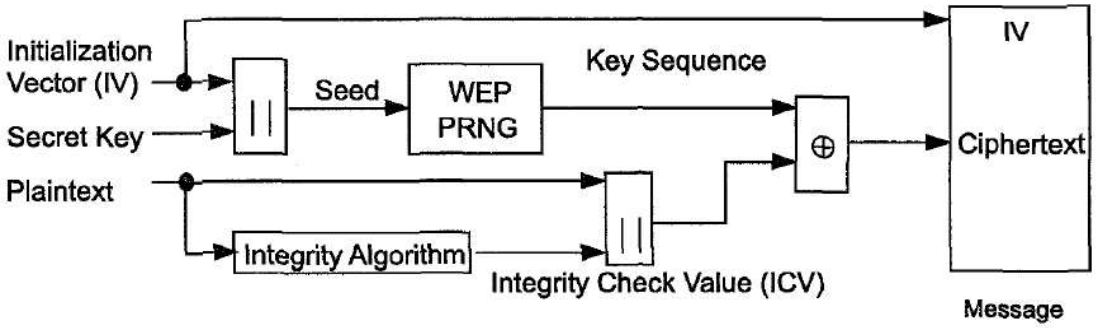

Figura 1: Diagramma della codifica WEP

Il primo passo dell'algoritmo prevede che la chiave segreta sia concatenata con un *initialization vector* (IV) e che la stringa risultante costituisca il seme di un generatore di numeri pseudocasuali, chiamato *pseudo-random number generator* (PRNG). L'output del PRNG è un *keystream* k la cui lunghezza è esattamente uguale a quella del messaggio che sarà trasmesso in rete.

Per proteggere dalla modifica del messaggio durante la trasmissione, gli si applica un algoritmo di controllo integrità. Nel caso specifico del WEP si usa CRC-32. Il risultato di questa operazione viene chiamato *Integrity Check Value* (ICV) e sarà concatenato al messaggio stesso.

Il processo di cifratura termina quindi calcolando lo XOR tra il *keystream* k ed il testo in chiaro concatenato con lo ICV.

Il messaggio finale, pronto per la trasmissione, è ottenuto unendo al testo cifrato lo IV iniziale in chiaro. È necessario che lo IV sia inviato in chiaro per permettere la decodifica al destinatario.

La decodifica di un messaggio ricevuto, schematizzata in figura 2, prevede una fase iniziale in cui si genererà lo stesso *keystream* k utilizzato per la codifica.

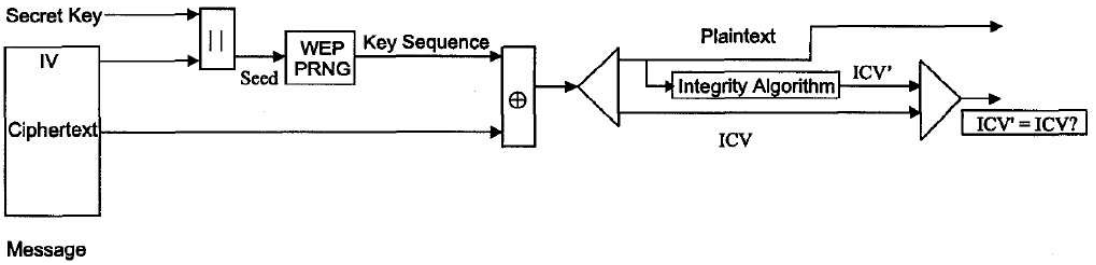

Figura 2: Diagramma di decodifica WEP

Questo avviene prendendo lo IV del messaggio ricevuto, concatenandolo alla chiave segreta ed infine utilizzando la stringa ottenuta come input del PRNG. Quindi si

calcolerà lo XOR tra il *keystream* ottenuto ed il messaggio cifrato. Infine la corretta decifratura deve essere verificata dall'algoritmo CRC-32 sul testo in chiaro recuperato. Si controlla se lo ICV contenuto nel messaggio ricevuto corrisponda esattamente allo ICV appena calcolato.

Lo standard 802.11 prevede che si utilizzi RC4 [6] come algoritmo di PRNG. Inoltre il protocollo permette di cambiare il valore dello IV di ogni messaggio in modo che RC4 usi una chiave diversa ogni volta.

2.3 RC4

RC4 [6] è stato sviluppato nel 1987 da Ron Rivest per RSA Data Security, Inc.

Per sette anni l'algoritmo è stato proprietario e i dettagli venivano resi disponibili solo dopo la sottoscrizione di un accordo di non divulgazione. Nel settembre del 1994, è stato inviato in modo anonimo il codice sorgente dell'algoritmo sulla mailing list di Cypherpunks, diffondendosi velocemente in tutto il mondo. Alcuni lettori già in possesso del codice originale hanno confermato la compatibilità con il codice online.

L'algoritmo RC4 è di facile descrizione: esso utilizza una *substitution box* ad 8 bit: $S_0, S_1, \ldots, S_{255}$. La quale è inizializzata linearmente: $S_0 = 0, \ldots, S_{255} = 255$. Oltre all'array S è necessario un altro array K di 256 byte che viente inizializzato con i valori della chiave, ripetendola nel caso la lunghezza fosse inferiore. L'ultima fase dell'inizializzazione prevede l'esecuzione di questo algoritmo, che permette di distribuire i valori di S:

$j = 0$
for $i = 0$ to 255 :
    $j = (j + S_i + K_i)$ mod 256
    swap S_i and S_j

Per generare un qualsiasi byte del *keystream*, si effettuano le seguenti operazioni:

$i = (i + 1)$ mod 256
$j = (j + S_i)$ mod 256
swap(S_i, S_j)
$t = (S_i + S_j)$ mod 256
$k = S_t$

Come si può notare, sono necessari due contatori, indicati con i e j ed inizializzati a zero.

Per produrre codice cifrato è sufficiente calcolare lo XOR fra il byte k ed il corrispondente byte del testo in chiaro, così come per ritornare al testo in chiaro si farà lo XOR fra k e il codice cifrato.

La *substitution box* si evolve lentamente con l'uso: i garantisce che ogni elemento sia variato, mentre j assicura che gli elementi cambino in modo casuale.

La versione ufficiale di RC4 precedentemente descritta è a 8 bit. Mentre si potrebbe pensare di definirne una versione con *substitution box* a 16 bit.

L'esecuzione di una cifratura utilizzando questo algoritmo risulta circa dieci volte più veloce rispetto ad una eseguita con l'algoritmo DES.

La RSA Data Security ha dichiarato che l'algoritmo RC4 è immune da criptanalisi, sebbene tale dichiarazione non sia stata verificata, in quanto non ci sono risultati pubblici, il fatto che RC4 possa presentarsi in 2^{1700} possibili stati differenti costituisce una buona garanzia.

2.4 Gli obiettivi del WEP

Come si è visto nel paragrafo 1.3, i principali problemi delle reti wireless possono essere sostanzialmente suddivisi in tre categorie. Il protocollo WEP è stato specificatamente progettato per rafforzare questi tre aspetti [7]:

Riservatezza L'obiettivo fondamentale del WEP è prevenire le intercettazioni casuali. La riservatezza è ottenuta codificando con l'algoritmo RC4 i pacchetti inviati. Sia l'utente finale che l'Access Point dispongono della medesima chiave, necessaria per riportare i messaggi cifrati in testo in chiaro.

Access Control Il secondo obiettivo del protocollo è di proteggere l'accesso non autorizzato alla rete wireless. Lo standard 802.11 include infatti una caratteristica che permette di scartare tutti i pacchetti che non sono adeguatamente codificati con WEP.

Data Integrity Un obiettivo correlato è di prevenire l'alterazione dei messaggi trasmessi, il campo di integrity checksum ICV è incluso proprio per questo scopo.

Attualmente ci sono due metodi di implementazione del WEP: il metodo WEP classico previsto dallo standard 802.11 ed una versione estesa sviluppata da molti produttori per permettere l'uso di chiavi più lunghe. Lo standard WEP specifica l'uso di chiavi da soli 40 bit, a causa delle limitazioni che erano imposte da leggi americane sulla crittografia al momento della stesura dello standard. La versione estesa di WEP prevede l'uso di chiavi cosiddette a 128 bit, anche se la dimensione reale è di 104 bit, in modo da rendere più arduo un attacco a forza bruta.

3 Debolezze del protocollo WEP

Il protocollo WEP è stato progettato come strumento per salvaguardare la riservatezza dei pacchetti trasmessi in rete, seguendo lo standard 802.11. Il WEP prevede che ogni pacchetto inviato sia codificato utilizzando una chiave segreta, di 40 bit o 104 bit, preceduta da un *initialization vector* IV di 24 bit e specifico di ogni pacchetto trasmesso. La stringa così ottenuta è utilizzata come chiave per l'algoritmo RC4 [6].

3.1 Vulnerabilità dell'algoritmo RC4

Nel 2001, i ricercatori Scott Fluhrer, Itsik Mantin e Adi Shamir hanno dimostrato l'esistenza di un'importante debolezza nell'algoritmo di *key scheduling* di RC4 [8].

Essa riguarda l'esistenza di larghe classi di chiavi deboli, in cui una piccola parte della chiave segreta determina un grande numero di bit di output dell'algoritmo di *key scheduling*.

La debolezza si manifesta quando parte della chiave utilizzata dall'algoritmo di *key scheduling* viene esposta ad un attaccante. Nel caso del WEP, questa situazione si verifica sempre. Infatti la parte iniziale della chiave è costituita dallo IV che è sempre trasmesso in chiaro in ogni pacchetto. L'attaccante può quindi individuare la chiave segreta intercettando gli IV di numerosi e differenti pacchetti.

Questo tipo di attacco può essere utilizzato qualunque sia la lunghezza dello IV; si presta quindi ad essere effettuato sia per attaccare il WEP, che usa un IV di 24 bit, ma anche contro la sua proposta di estensione, a volte chiamata WEP2, che invece utilizza un IV di 128 bit.

Scelta dello IV Siccome lo standard WEP non specifica come lo IV debba essere scelto, i produttori di schede 802.11 hanno scelto diversi modi per generarne la sequenza. La maggior parte di essi utilizza uno di questi tre metodi: contatore, selezione casuale o *value-flipping*, cioè l'utilizzo alternato di due valori. La tecnica del *value-flipping*, se associata ad un'accurata scelta dei valori, può evitare la vulnerabilità relativa alle chiavi deboli sopra descritta. Ma contemporaneamente comporta un forte riutilizzo della stessa chiave e quindi non può essere considerato un metodo efficace per evitare altri tipi di attacchi. Le schede che utilizzano un contatore per la generazione dello IV sono sicuramente quelle che si prestano maggiormente all'attacco. In queste schede lo IV è incrementato di uno ad ogni pacchetto inviato, partendo dal valore 0 o da un valore casuale scelto all'accensione. Sia nel caso che si utilizzi un contatore, sia che si usino valori casuali dello IV, un attaccante si troverebbe in una situazione favorevole. Infatti avrebbe la garanzia di una buona distribuzione nell'uso delle chiavi deboli.

3.2 Vulnerabilità specifiche del protocollo WEP

Nello stesso anno in cui furono evidenziate le debolezze intrinseche nell'algoritmo di codifica RC4, i ricercatori Nikita Borisov, Ian Goldberg e David Wagner hanno mostrato altre vulnerabilità presenti nel protocollo WEP [7].

3.2.1 Il riutilizzo dello stesso keystream

WEP alla base del proprio sistema di sicurezza prevede l'utilizzo dell'algoritmo RC4, il quale appartiene alla categoria degli *stream chipher*.

Ogni algoritmo di questo tipo agisce espandendo una chiave segreta, nel caso del WEP si tratta di un IV pubblico concatenato ad una chiave segreta, in uno stream arbitrariamente lungo di bit pseudo-casuali. La codifica avviene poi calcolando lo XOR tra lo stream generato ed il testo in chiaro. La decodifica avviene seguendo le stesse operazioni: si genera uno stream identico al precedente ed eseguendone lo XOR con il testo codificato, si otterrà il testo in chiaro.

Un noto punto debole degli algoritmi di *stream chypher* consiste nel fatto che il codificare due messaggi con una stessa chiave (nel caso del WEP, con uno stesso IV e con la stessa chiave), può rivelare informazioni riguardo ad entrambi i messaggi.
Infatti, se definiamo i due messaggi codificati C_1 e C_2 come:

$$C_1 = P_1 \oplus \text{RC4}(v, k)$$
$$C_2 = P_2 \oplus \text{RC4}(v, k)$$

si ottiene che:

$$C_1 \oplus C_2 = (P_1 \oplus \text{RC4}(v, k)) \oplus (P_2 \oplus \text{RC4}(v, k)) = P_1 \oplus P_2$$

In altre parole, calcolando lo XOR tra i due messaggi cifrati si riesce ad eliminare l'effetto del *keystream*, ottenendo come risultato lo XOR tra i due messaggi in chiaro.

Questa proprietà può rendere possibile diversi tipi di attacchi: se infatti il testo in chiaro di uno dei due messaggi fosse conosciuto, allora si otterrebbe immediatamente anche il secondo. Più in generale, nelle situazioni reali, spesso i messaggi in chiaro hanno abbastanza ridondanza tra loro da permettere di risalire ad entrambi semplicemente analizzandone lo XOR che si ha a disposizione. Queste tecniche diventano notevolmente più semplici da applicare avendo a disposizione un elevato numero di pacchetti codificati con la stessa chiave.

Per evitare questi attacchi, WEP utilizza un IV che varia in ogni pacchetto trasmesso. In questo modo si cambia il processo di generazione del *keystream* in ogni messaggio. Lo IV è inviato in chiaro negli header del pacchetto in modo che il ricevente possa generare lo stesso *keystream* necessario per la decodifica. In questo modo però si espone lo IV ad eventuali attaccanti, anche se la parte rimanente della chiave segreta rimane sconosciuta.

La variazione dello IV in ogni pacchetto trasmesso è stata introdotta per evitare il riutilizzo della stessa chiave, ma purtroppo il protocollo WEP non riesce a raggiungere

questo scopo. Occorre infatti notare che lo standard WEP si limita a raccomandare, ma non ad obbligare, che lo IV sia cambiato in ogni pacchetto ed inoltre non indica alcun modo in cui tale variazione debba avvenire. A causa di questa mancanza molti produttori hanno realizzato schede che inizializzano lo IV a 0 per il primo pacchetto e lo incrementano di uno in ogni pacchetto successivo.

Un altro aspetto da considerare riguarda la lunghezza dello IV, essa è di soli 24 bit, alcuni studi hanno mostrato come un normale Access Point sottoposto ad elevato traffico esaurisca tutti gli IV disponibili in meno di mezza giornata. Altri costruttori, utilizzano invece IV scelti in modo casuale, ma in questo caso la situazione può addirittura peggiorare: a causa del cosiddetto "paradosso del compleanno" ci si può aspettare un riutilizzo dello stesso IV dopo la trasmissione di soli 5000 messaggi, cioè dopo solo pochi minuti.

3.2.2 Modifica dei messaggi

Il protocollo WEP prevede che all'interno di ogni pacchetto sia inserito un campo contenente il checksum calcolato con CRC-32, questo campo sarà criptato insieme al messaggio da trasmettere.

Il checksum CRC non è però sufficiente ad assicurare che un attaccante non possa in alcun modo modificare un messaggio inviato. CRC è stato infatti progettato per identificare errori casuali durante la trasmissione del messaggio, ma non è in grado di resistere ad attacchi effettuati da utenti malevoli. Questo tipo di vulnerabilità è addirittura ampliata dal fatto che il corpo del messaggio sia codificato utilizzando un algoritmo di *stream cypher*.

Per dimostrare questa debolezza bisogna considerare che il checksum è una funzione lineare del messaggio, questo significa che se c è la funzione di checksum, si ha che $c(x \oplus y) = c(x) \oplus c(y)$. Questa è una proprietà generale di tutte le funzioni CRC.

Per mostrare come possa essere sfruttato da un attaccante, consideriamo un messaggio codificato C che viene intercettato e chiamiamo M il corrispondende messaggio in chiaro, si ha che

$$C = \text{RC4}(v, k) \oplus \langle M, c(M) \rangle$$

L'attaccante può modificare il messaggio M in un nuovo messaggio M', semplicemente facendo lo XOR tra il messaggio cifrato intercettato C ed un nuovo messaggio Δ tale che $M' = M \oplus \Delta$

Si consideri infatti la seguente equazione:

$$\begin{aligned} C' &= C \oplus \langle \Delta, c(\Delta) \rangle \\ &= \text{RC4}(v, k) \oplus \langle M, c(M) \rangle \oplus \langle \Delta, c(\Delta) \rangle \\ &= \text{RC4}(v, k) \oplus \langle M \oplus \Delta, c(M) \oplus c(\Delta) \rangle \\ &= \text{RC4}(v, k) \oplus \langle M', c(M \oplus \Delta) \rangle \\ &= \text{RC4}(v, k) \oplus \langle M', c(M') \rangle \end{aligned}$$

Essa mostra la validità del messaggio modificato. Infatti il destinatario decodificherà il messaggio e troverà un checksum corretto per M' e considererà quindi valido quanto avrà ricevuto.

Si noti che questo attacco può essere applicato anche senza conoscere nessuna parte del messaggio in chiaro M. L'attaccante deve solamente intercettare il messaggio cifrato C e decidere la differenza con l'originale, cioè Δ. Questo implica che si possano effettuare modifiche arbitrarie su un pacchetto senza che nessuno riesca ad identificare il cambiamento. La protezione dell'integrità dei dati è uno degli obiettivi che si era prefisso il protocollo WEP, ma il checksum adottato non permette assolutamente di raggiungerlo.

3.2.3 Invio non autorizzato di messaggi

L'utilizzo di checksum CRC all'interno del protocollo WEP comporta, oltre alla potenziale modifica di un messaggio trasmesso, anche la possibilità di inviare messaggi in una rete senza essere autorizzati ad accedervi.

La funzione di checksum non è infatti una funzione crittografica. Conseguentemente ogni attaccante che conosca il messaggio originale può calcolarne il checksum in modo autonomo. Quindi se riuscisse ad ottenere un messaggio cifrato ed il corrispondente messaggio in chiaro potrebbe, come si è visto nel paragrafo 3.2.1, ottenerne il relativo *keystream*. Il quale potrebbe così servire per creare nuovi pacchetti semplicemente riutilizzando lo stesso IV. Questa possibilità è dimostrata dalla seguente equazione in cui P è il messaggio in chiaro e C è il corrispondente cifrato:

$$P \oplus C = P \oplus (P \oplus \text{RC4}(v, k)) = \text{RC4}(v, k)$$

Avendo ottenuto il valore del *keystream* RC4(v, k), lo si può utilizzare per creare qualsiasi altro messaggio cifrato:

$$C' = \langle M', c(M') \rangle \oplus \text{RC4}(v, k)$$

È necessario utilizzare sempre lo stesso IV in ogni pacchetto, ma dal momento che il protocollo WEP non indica alcun modo su come esso debba variare, lo si può usare senza che questo crei nessun tipo di allarme da parte del destinatario. Una volta ottenuto un *keystream* lo si potrà utilizzare in modo indefinito ed in questo modo si riuscirà ad aggirare l'intero meccanismo di *Access Control* previsto dal WEP.

3.2.4 Falsificare l'autenticazione

Un caso particolare dell'invio non autorizzato dei messaggi può essere utilizzato per aggirare il meccanismo di autenticazione previsto dal WEP.

Questo meccanismo è utilizzato dagli Access Point per autorizzare gli host mobili prima che formino un'associazione con esso. Infatti il procedimento previsto dal

WEP indica che un Access Point, dopo aver ricevuto una richiesta di autenticazione da parte di un host mobile, gli invii un messaggio in chiaro di 128 byte casuali chiamato *challenge*. L'host mobile risponderà con lo stesso *challenge* dopo averlo codificato usando WEP. L'autenticazione avrà successo se la decodifica corrisponderà esattamente al messaggio inviato inizialmente dall'Access Point. La capacità di poter generare una versione codificata correttamente è considerata una dimostrazione di possesso della chiave WEP.

Come descritto però nel precedente paragrafo 3.2.3, è possibile inviare in una rete wireless dei pacchetti codificati correttamente senza dover conoscere la chiave WEP. Infatti è necessaria solamente la conoscenza di un messaggio in chiaro e del corrispondente codificato. Tale coppia di messaggi può essere facilmente ottenuta monitorando una legittima sequenza di autenticazione. In questo modo diventa estremamente semplice poter ricavare il *keystream* utilizzato. Inoltre, dato che i messaggi di autenticazione sono tutti della stessa lunghezza, il *keystream* ottenuto potrà essere usato per codificare un qualsiasi *challenge* ricevuto.

Questo tipo di attacco, oltre ad essere stato studiato da Borisov, Goldber e Wagner, è anche stato sviluppato in modo indipendente dai ricercatori Arbaugh, Shankar e Wan [9].

3.2.5 Decodifica mediante redirezione dei pacchetti

La possibilità di modificare i pacchetti inviati in una rete wireless può essere sfruttata da un attaccante per decodificarli.

L'idea si basa sul fatto che anche se l'attaccante non conosca la chiave WEP, in una rete ci sarà sempre almeno l'Access Point che può decodificare i pacchetti. Si cercherà quindi di fare in modo che, modificando opportunamente alcuni pacchetti, l'Access Point li decodifichi e li trasmetta ad un host sotto il controllo dell'attaccante.

L'attacco consiste in un "reindirizzamento IP" ed è possibile in tutte le reti wireless in cui l'Access Point sia anche un router con accesso ad Internet. Questa situazione è sicuramente molto comune.

L'attaccante utilizzerà le tecniche descritte nel paragrafo 3.2.2 per cambiare l'indirizzo destinazione di un pacchetto e costringere così l'Access Point a reindirizzarlo verso un host sotto controllo.

Per effettuare una modifica efficace è necessario conoscere il reale indirizzo di destinazione del pacchetto, ma questa operazione non costituisce una difficoltà. Infatti la maggior parte dei pacchetti sarà diretta verso la rete wireless stessa, i cui indirizzi sono facilmente identificabili.

L'unica difficoltà consiste nel mantenere un valore di *checksum* del pacchetto valido, anche dopo la modifica dell'indirizzo destinazione. In questo caso si possono utilizzare diversi accorgimenti per poter ottenere *checksum* validi. Il più semplice di essi consiste nel mantenerne il valore, cercando ad esempio di modificare anche l'indirizzo sorgente del pacchetto.

4 Esperimenti per provare la debolezza del WEP

4.1 Il primo esperimento pubblico

Poco tempo dopo la pubblicazione dell'articolo di Fluhrer, Mantin e Shamir riguardante la scoperta di vulnerabilità nell'algoritmo RC4 [8], fu condotto un primo esperimento, da parte dei ricercatori Stubblefield, Ioannidis e Rubin, per verificare concretamente quanto scoperto [10].

L'esperimento condotto si poneva principalmente due obiettivi: prima di tutto si voleva verificare che l'attacco descritto potesse essere funzionante anche nel "mondo reale" e non solo in teoria. In secondo luogo si voleva capire quanto potesse essere facile ed economico condurre questo tipo di attacco.

La prima fase dell'esperimento ha previsto la simulazione di un attacco a RC4. Dopo appena due ore dedicate alla scrittura del codice, si fece partire la simulazione che ne mostrò l'effettivo funzionamento. La parte che invece richiese meggiore tempo riguardò la cattura dei pacchetti codificati con WEP. A questo scopo fu acquistata una scheda di rete dotata di chipset Prism II del costo di 100 $, la quale permetteva di effettuare molte computazioni via software e inoltre consentiva la cattura di pacchetti grezzi.

L'ultima parte dell'esperimento consistette nell'individuare il valore del primo byte del messaggio in chiaro. Questa operazione fu estremamente semplice, infatti il traffico maggiormente presente nella rete locale era di tipo IP e ARP. I quali erano però tutti incapsulati con un header 802.2, come previsto dal protocollo SNAP [11]. Tale header iniziava sempre con il valore `0xAA` ed esso costituiva il primo byte del messaggio in chiaro.

Avendo scoperto il valore del primo byte, si usò il seguente algoritmo per individuare la chiave WEP utilizzata durante l'esperimento:

RecoverWEPKey()
    $Key[0 \ldots KeySize] = 0$
    for $KeyByte = 0 \ldots KeySize$
        $Counts[0 \ldots 255] = 0$
        foreach packet $\rightarrow P$
            if $P.IV \in \{(KeyByte + 3, \texttt{0xFF}, N) \mid N \in \texttt{0x00} \ldots \texttt{0xFF}\}$
                $Counts[\text{SimulateResolved}(P, Key, KeyByte)] \mathrel{+}= 1$
        $Key[KeyByte] = \text{IndexOfMaximumElement}(Counts)$
    return Key

SimulateResolved$(P, Key, KeyByte)$

$K = P.IV \cdot Key$

for $i = 0 \ldots N - 1$

$S[i] = i$

for $i = 0 \ldots KeyByte$

$j = j + S[i] + K[i \bmod l]$

swap$(S[i], S[j])$

return $S_{B+2}^{-1}[P.Out] - j_{B+2} - S_{B+2}[B + 3]$

Con questo esperimento, Stubblefield, Ioannidis e Rubin dimostrarono l'effettiva applicabilità dell'attacco all'algoritmo RC4, mostrando come fosse possibile ricavare la chiave WEP utilizzata in una determinata rete wireless. Riuscirono inoltre a condurre l'esperimento con mezzi economici ed alla portata di chiunque, potendo quindi concludere che il protocollo WEP previsto dallo standard 802.11 fosse completamente insicuro.

4.2 Airsnort

Airsnort [12] è un programma, rilasciato con licenza GPL [13], che sfrutta le vulnerabilità nell'algoritmo di *key scheduling* di RC4 [8] permettendo di individuare la chiave WEP utilizzata in una determinata rete.

Airsnort agisce monitorando in modo passivo le trasmissioni in una rete WLAN. Una volta ottenuti abbastanza pacchetti codificati con WEP (mediamente ne sono necessari tra i 5 ed i 10 milioni) è in grado di stimare la chiave in circa un secondo di computazione.

La "cattura" dei pacchetti avviene attraverso un processo che ne prevede il passaggio attraverso due filtri. Il primo di essi si occupa di individuare i pacchetti che non sono codificati con WEP e di scartarli in quanto non necessari. Nel secondo, invece, si controlla lo IV utilizzato per codificare i pacchetti; in questo stadio sono scartati tutti quelli che utilizzano un IV che non è considerato debole [8], poiché essi non sono utili per effettuare l'attacco. Sono scartati anche tutti i pacchetti non contenenti dati, ad eccezione di alcuni utilizzati per ricavare le informazioni sul *SSID* del quale fa parte l'Access Point.

I tentativi di *cracking* della chiave WEP sono effettuati parallelamente alla cattura dei pacchetti. Airsnort tenta di ricavare sia una chiave a 40 bit sia una a 128 bit ogni dieci pacchetti con IV debole. L'attacco utilizzato è di tipo probabilistico, quindi la stima migliore potrebbe non essere quella corretta. Con un numero limitato di dati catturati ed abbastanza potenza di CPU, è possibile effettuare ricerche più esautive. Questa ricerca della chiave prevede la visita di un albero n-ario, la cui ampiezza è regolata attraverso l'interfaccia grafica di Airsnort e che ha profondità 5 per quanto

riguarda i tentativi con le chiavi a 40 bit e profondità 13 per le chiavi a 128 bit. Impostando un'ampiezza n, si indica ad Airsnort di provare gli n valori più probabili utilizzando le statistiche ricavate dagli IV collezionati. Impostare valori grandi per l'ampiezza causa molta lentezza nell'effettuare i tentativi di *cracking*, per cui si ritiene che i valori maggiormente indicati per n siano 3 per le chiavi a 40 bit e 2 per quelle a 128 bit.

Il numero di pacchetti, con IV debole, necessari per poter individuare la chiave, dipende essenzialmente da due fattori. Il primo di essi è la fortuna poiché l'attacco è di tipo probabilistico, mentre il secondo è la lunghezza della chiave. Mediamente sono necessari circa 1500 pacchetti per chiavi a 128 bit e circa la metà per chiavi a 40 bit; anche se alcune chiavi sono più resistenti a questo attacco e richiederanno un numero decisamente maggiore di pacchetti.

Nel momento in cui Airsnort supponga di aver stimato correttamente la chiave, proverà a decodificare un pacchetto scelto casualmente. Nel caso in cui il valore di checksum sia corretto, la chiave WEP sarà stata identificata e verrà visualizzata all'utente.

4.2.1 Il nostro esperimento

Per dimostrare la facilità con cui può essere realizzato un attacco ad una WLAN utilizzando Airsnort, abbiamo deciso di effettuare un esperimento avviando il programma all'intero di una rete domestica, per verificare se si riuscisse ad ottenere la chiave WEP.

Il test è stato effettuato su un sistema GNU/Linux. Per poter funzionare è stato necessario utilizzare una scheda 802.11 che supportasse la modalità "Monitor", in cui la scheda agisse come monitor passivo e si limitasse a ricevere i pacchetti inviandoli in modo grezzo al kernel del sistema operativo. Nel nostro caso è stata utilizzata una scheda 802.11b con chipset Prism II e i driver Host AP [14], scelti per il loro supporto alle *wireless extension* Linux, potendo così utilizzare i tool standard inclusi in tutte le distribuzioni, e per la capacità di operare in "Monitor mode".

L'esperimento non è stato condotto all'interno di una rete wireless "reale", ma è stato svolto utilizzando un altro host che inviasse continuamente pacchetti in modo da simulare una rete satura. L'invio continuo di pacchetti è stato possibile utilizzando il modulo packet generator (*pktgen.o*) presente all'interno del kernel Linux.

Dopo aver iniziato l'invio dei pacchetti, nel computer attaccante è stato avviato Airsnort. Il programma si presenta con un'interfaccia grafica che permette di regolarne la configurazione e consente di visualizzare costantemente la situazione dell'attacco. In figura 3 è possibile vedere come si presenta il programma subito dopo l'avvio.

Si può notare che Airsnort è in grado di individuare autonomamente il nome del *SSID* utilizzato nella rete wireless. Alcuni produttori di Access Point, per migliorare il meccanismo di Access Control dei propri prodotti, vi aggiungono la possibilità di non inviare periodicamente le informazioni relative al proprio *SSID*. Purtroppo, come si è visto, è possibile ottenere tali informazioni molto velocemente ed in modo

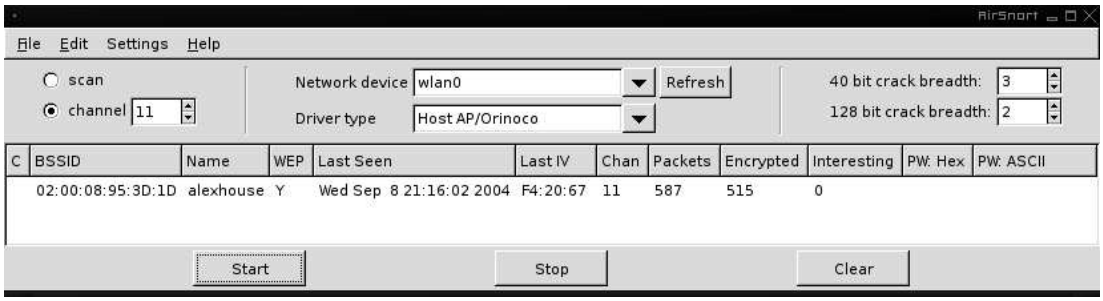

Figura 3: Avvio di Airsnort

estremamente semplice. La funzionalità aggiunta dai produttori si rivela quindi essere totalmente inutile e, poiché contribuisce a generare una falsa sensazione di sicurezza, anche estremamente dannosa.

Poco dopo l'avvio, Airsnort individua i primi pacchetti che abbiano utilizzato un IV considerato debole. Il numero totale di questo tipo di pacchetti è mostrato dall'interfaccia del programma, come si può vedere in figura 4. Essi vengono tutti

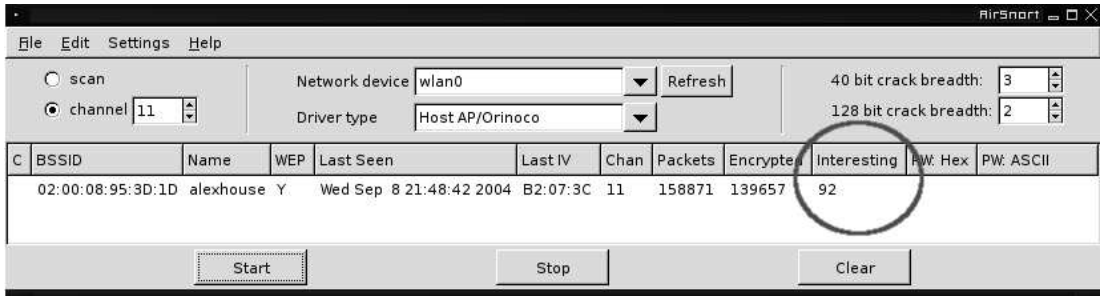

Figura 4: Rilevamento pacchetti con IV debole

memorizzati e sono utilizzati per effettuare l'attacco vero e proprio.

L'attacco è stato totalmente passivo, il computer che eseguiva Airsnort è stato lasciato acceso durante la notte, in modo che potesse individuare un numero sufficiente di pacchetti codificati utilizzando IV deboli. Dopo averne collezionati più di 1500, senza che si individuasse la chiave, si è deciso di aumentare l'ampiezza dell'albero utilizzato da Airsnort, portandolo da 3 a 15 ed aumentando quindi le computazioni effettuate successivamente.

Nella mattinata successiva, dopo aver identificato circa 2800 pacchetti deboli è stato possibile scoprire la chiave WEP utilizzata: *"linux"*; si noti infatti la figura 5 che mostra la chiave WEP da poco individuata.

La durata dell'esperimento, dall'avvio di Airsnort alla sua identificazione della chiave, è stata di circa otto ore ed è stato effettuato utilizzando apparecchiature alla portata di chiunque: un notebook di media potenza ed una comune scheda di rete wireless 802.11b. Anche il software utilizzato è estemamente semplice da

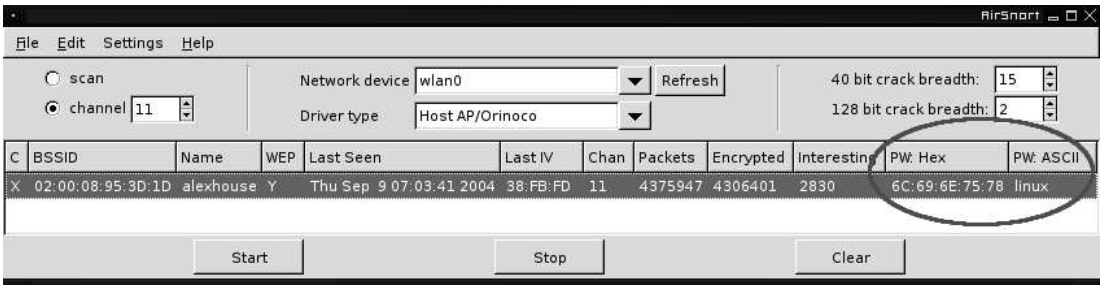

Figura 5: Identificazione della chiave WEP

usare e, essendo rilasciato con licenza GPL, è disponibile per chiunque senza alcuna limitazione.

Risulta chiaro quindi come un tale esperimento possa essere riprodotto senza problemi in qualsiasi situazione reale. Anche il tempo necessario è da ritenere breve, considerando che lo stato di Airsnort può essere salvato su file in modo da riprendere l'attacco in un momento successivo. Si può quindi affermare che l'algoritmo di codifica WEP fornisce una protezione decisamente debole e che, nelle situazioni in cui la sicurezza delle proprie reti rivesta una qualsiasi importanza, debbano essere considerate unicamente delle soluzioni alternative.

4.3 WEPCrack

Oltre al programma Airsnort [12] che, come descritto nel paragrafo 4.2.1, abbiamo utilizzato per condurre il nostro esperimento, esiste anche il programma WEPCrack [15] che presenta funzionalità del tutto simili.

WEPCrack è un tool, rilasciato con licenza GPL [13], che permette di individuare la chiave WEP utilizzata in una determinata rete. Anch'esso agisce implementando l'attacco descritto da Fluhrer, Mantin e Shamir [8].

Mentre Airsnort è il programma che maggiormente ha catturato l'attenzione, WEPCrack è stato il primo, con il codice pubblicamente disponibile, che ha permesso di dimostrare l'attacco contro l'algoritmo RC4. È stato annunciato sulla mailing list BugTraq di SecurityFocus [16]. L'esperimento realizzato da Stubblefield [10], descritto nel paragrago 4.1, fu il primo ad essere annunciato pubblicamente ma non ne fu mai rilasciato il codice sorgente.

5 Proposte ufficiali per sostituire il protocollo WEP

5.1 Il protocollo 802.11i

Per correggere i noti problemi del WEP, il Working Group relativo al protocollo 802.11 ha istituito un nuovo gruppo di lavoro, chiamato “Task Group i”, il cui compito è la definizione di un nuovo standard. Il nuovo gruppo attualmente non è ancora giunto alla stesura del protocollo definitivo, che prenderà il nome di 802.11i [17].

Da ciò che si può notare dai *draft* pubblicati, probabilmente lo standard sarà composto da due differenti protocolli. Il primo, chiamato *Temporal Key Integrity Protocol* (TKIP), avrà come obiettivo una veloce adozione da parte dei produttori, in modo da correggere in tempi brevi i problemi di sicurezza. Il secondo, chiamato *Counter Mode CBC-MAC Protocol* (CCMP), sarà invece una soluzione a lungo termine ed adotterà sistemi sicurezza maggiormenti evoluti.

5.1.1 Temporal Key Integrity Protocol (TKIP)

Siccome la maggior parte dei sistemi IEEE 802.11 implementano il protocollo WEP a livello hardware [18], per risolverne le vulnerabilità, “il Task Group i” ha definito il *Temporal Key Integrity Protocol* (TKIP). Questo nuovo protocollo potrà funzionare anche con l’hardware esistente, inoltre la sua utilizzazione prevede solamente un aggiornamento del firmware e dei driver.

TKIP è comunque da intendersi solo come una soluzione temporanea. La sua effettiva applicabilità sull’hardware attuale dipende da diversi fattori. Il sistema installato deve infatti possedere un firmware aggiornabile via software. Inoltre non deve essere variata l’implementazione WEP della scheda di rete, nel caso in cui essa avvenga in hardware. È infine necessario minimizzare il degrado delle performance dovuto all’utilizzo di tecniche più complesse.

Per risolverne i problemi individuati, TKIP modifica direttamente il protocollo WEP, ma non lo sostituisce in modo totale. Cerca infatti di utilizzare nuove tecniche, per raggiungere gli scopi che il WEP si prefiggeva, ma eliminandone le vulnerabilità.

Message Integrity Code Per garantire l’integrità dei messaggi e prevenirne le alterazioni, è previsto l’utilizzo di un nuovo algoritmo detto *Message Integrity Code* (MIC), a volte chiamato anche *Michael*. Esso è una funzione che prende in input una chiave a 64 bit, l’indirizzo fisico della scheda di rete, l’indirizzo del destinatario ed il messaggio. Come output restituisce il messaggio concatenato ad una stringa generata. Nel caso fosse necessario, procederà anche alla frammentazione del pacchetto prima che sia trasmesso.

Per-packet Key Il WEP, prevedendo la concatenazione tra la chiave base ed un IV di 24 bit, si è dimostrato vulnerabile all’attacco individuato da Fluhrer, Mantin e Shamir. Per difendere il sistema da questo tipo di attacco, TKIP introduce un nuovo

procedimento, il quale permette di ottenere una chiave specifica per ogni pacchetto da inviare (*per-packet key*). La funzione che porta a questo risultato ha come input l'indirizzo fisico del dispositivo di rete ed il numero di sequenza a 48 bit del pacchetto [19]. L'utilizzo di 48 bit, invece dei 24 bit dello IV usato dal WEP, riesce a limitare molto la possibilità di riutilizzo della chiave. L'output della funzione genera una nuova chiave a 128 bit per ogni pacchetto, da utilizzare per effettuare la codifica con WEP. Dato che si utilizza un numero di sequenza, il quale si incrementa dopo l'invio di ogni pacchetto, risulta che la chiave è usata per un'unica trasmissione. Il numero di sequenza è anche utile come difesa contro certi tipi di attacchi: il destinatario di un pacchetto non ne accetterà se dovessero avere un numero di sequenza uguale o più piccolo dei precedenti ricevuti.

Come si è visto, TKIP richiede due chiavi distinte: una a 128 bit, specifica di ogni pacchetto, ed un'altra a 64 bit impiegata da MIC. Il nuovo protocollo richiede che queste chiavi siano rinnovate ogni volta che si effettui un'associazione con un Access Point. Per questo motivo è stato adottato lo standard 802.1X [20], descritto in dettaglio nel paragrafo 5.2, in modo da fornire un metodo efficace con il quale effettuare queste autenticazioni.

È interessante notare come TKIP preveda comunque di codificare i pacchetti usando WEP alla fine delle operazioni sopra descritte.

5.1.2 Counter Mode CBC-MAC Protocol (CCMP)

Come il TKIP, anche il protocollo *Counter Mode CBC-MAC Protocol* (CCMP) ha come obiettivo la risoluzione dei problemi di sicurezza del WEP, senza però essere vincolato all'utilizzo dell'hardware preesistente.

Una delle maggiori novità rispetto a TKIP e WEP consiste nell'aver adottato un nuovo algoritmo di crittografia, CCMP abbandona infatti l'uso di RC4 per codificare i dati mediante l'algoritmo *Advanced Encryption System* (AES) [21].

Questo protocollo, che si propone come soluzione a lungo termine, presenta molte caratteristiche in comune con la soluzione a breve termine TKIP. Ma CCMP ha il vantaggio di essere indipendente dalla scelta dell'hardware potendo così impiegare soluzioni più eleganti.

Esattamente come in TKIP, anche CCMP impiega un numero di sequenza a 48 bit, caratteristica che assicura un tempo di vita della chiave AES sicuramente superiore di ogni reale associazione con un Access Point.

Il semplice uso di AES, che appartiene alla famiglia dei *block chiper*, al posto dell'algoritmo RC4, porta notevoli miglioramenti. Con esso non è più necessario l'uso di chiavi "per pacchetto" ed inoltre, la stessa chiave è utilizzata sia per garantire riservatezza, sia per garantire l'integrità dei dati.

5.2 Protocollo 802.1X

IEEE 802.1X [20] è un protocollo di autenticazione *port-based* studiato specificatamente per proteggere una qualsiasi rete ethernet.

Si pone infatti l'obiettivo di evitare gli accessi non autorizzati in reti aperte, come ad esempio quelle presenti nei campus universitari, dove qualsiasi presa a muro attiva rappresenta un punto d'accesso nell'infrastruttura. Il protocollo 802.1X consente alla porta di rimanere attiva, ma richiede autenticazione prima che un utente riceva accesso completo.

Questo concetto di porta fisica ultimamente è stato esteso alle reti wireless [22]. Il fatto che in esse si trasmetta attraverso l'aria, rende la situazione equivalente all'avere a disposizione una presa a muro ogni volta che si entri all'interno della portata di un Access Point.

L'autenticazione dello 802.1X è richiesta sia nel momento in cui un host entri per la prima volta nella rete, sia ad intervalli periodici per verificare che l'host non sia stato rimosso o che altri non abbiano preso il suo posto.

802.1X ha il vantaggio di non aggiungere overhead alla trasmissione dei pacchetti. Questa implementazione leggera è importante perché non porta effetti indesiderati sulla capacità relativamente bassa delle reti wireless.

Per effettuare le autenticazioni, l'Access Point si appoggia ad un server backend via *Remote Authentication Dial-In User Service* (RADIUS), il quale si occupa della verifica. Una volta che i processi di autenticazione si siano completati, il server di autenticazione manderà un messaggio all'Access Point comunicando che l'host è stato autorizzato e che può essere garantito l'accesso alla rete.

L'autenticazione vera e propria è eseguita per mezzo di *Extensible Authentication Protocol* (EAP), il quale possiede un'importante caratteristica. Infatti, anziché specificare un meccanismo di autenticazione fisso, EAP fornisce una piattaforma estendibile. Se dovesse essere scoperto un problema di sicurezza nel meccanismo di autenticazione utilizzato, questo potrebbe essere sostituito con uno più robusto.

Mentre TKIP e CCMP non sono ancora completamente implementati dai principali produttori hardware, il supporto a 802.1X è già disponibile per i principali sistemi operativi.

5.3 Wi-Fi Protected Access (WPA)

Poco tempo dopo la scoperta delle vulnerabilità che affliggono il WEP, la Wi-Fi Alliance, organizzazione che si occupa di promuovere lo sviluppo e la diffusione delle reti wireless, ha intrapreso una collaborazione con IEEE per poter trovare una soluzione che fosse interoperabile e che potesse essere disponibile entro un anno circa. I risultati di questo lavoro sono conosciuti con il nome di *Wi-Fi Protected Access* (WPA) [23].

I principali obiettivi di WPA, scelti per fornire miglioramenti al livello di sicurezza, sono di definire un metodo di autenticazione degli utenti e di rendere più efficace la

codifica dei messaggi. Inoltre WPA prevede anche la possibilità di essere aggiornabile via software.

Per la codifica dei messaggi, WPA si affida al protocollo TKIP, descritto nel paragrafo 5.1.1, mentre per l'autenticazione è stato scelto di utilizzare il protocollo EAP assieme allo standard 802.1X

Queste scelte sono state effettuate in modo da rendere WPA compatibile con quello che diventerà lo standard 802.11i. Le specifiche di WPA possono essere infatti considerate un sottoinsieme di quanto previsto dall'attuale *draft*.

6 Altre soluzioni proposte

Dopo aver ampiamente discusso riguardo ai problemi di sicurezza del WEP ed averne analizzato alcune soluzioni, in questo capitolo si presenterà una breve analisi di altre soluzioni proposte.

6.1 Syncronized Random Numbers for Wireless Security (SPRiNG)

SPRiNG [24] è un semplice protocollo per proteggere le comunicazioni Point-to-Point. Come nel WEP, gli obiettivi di SPRiNG sono l'autenticazione, la riservatezza e l'integrità dei dati.

Come in molti protocolli di sicurezza, il WEP raggiunge questi obiettivi attraverso la cifratura e il controllo di integrità. L'autenticazione si ottiene invece implicitamente, infatti se un messaggio supera il controllo di integrità significa che debba essere stato codificato usando la chiave segreta.

Il protocollo SPRiNG, al contrario, agisce in modo differente. Invece di delegare i meccanismi di Access Control ad altri sistemi, prevede l'utilizzo di uno specifico meccanismo di autenticazione.

6.2 Variable Encrypting Function (VEF)

Il *Variable Encrypting Function* (VEF) [25] è basato sull'incremento della "diffusione" caratteristica, usando un algoritmo *block chaining* a livello application, rendendo dipendente il carattere del testo cifrato al corrente testo in chiaro e al precedente carattere del testo in chiaro. Questo meccanismo consiste di una tabella che ha 128 permutazioni dai numeri 0 a 127. Queste permutazioni sono indicizzate in un ordine particolare.
La corrispondente permutazione è selezionata in base al valore ASCII del precedente carattere del testo in chiaro. Il carattere cifrato è ottenuto usando questa permutazione e il corrente carattere ASCII del testo in chiaro. Includendo la dipendenza del precedente carattere si assicura che lo "sniffing" di frame durante le trasmissioni, dovrebbe essere inefficace.

6.3 Multipath Ad-Hoc Routing

È stato proposto un nuovo meccanismo di *routing multipath* [26] per combattere i problemi di sicurezza, presenti a livello data-link, delegandone la gestione al livello di rete superiore.

Questo approccio non richiede che l'applicazione usi sofisticate tecniche di crittografia, le quali potrebbero essere troppo pesanti per i dispositivi mobili.

Consiste nell'instradare i pacchetti inviati secondo differenti percorsi, in modo da diminuire le possibilità di intercettazione.

6.4 Virtual Private Network (VPN)

La tecnica delle *Virtual Private Network* (VPN) non è stata studiata appositamente per trovare soluzione ai problemi del WEP, ma esiste da diverso tempo come tecnologia per trasmettere in modo sicuro dei dati attraverso reti non affidabili.

Questa caratteristica è stata considerata molto importante per evitare le vulnerabilità presenti nelle reti wireless. La scoperta dell'inefficacia del protocollo WEP ha portato alcuni ricercatori a proporre di considerarle sempre come totalmente insicure ed accessibili a chiunque, per poi cercare soluzioni ad un livello superiore [27].

VPN opera infatti ad un livello più alto di quello data-link utilizzato dal WEP e permette la creazione di tunnel crittografati tra due host.

Riferimenti bibliografi ci

[1] IEEE. *IEEE Std 802.11-1997*, Nov 1997. IEEE Std 802.11-1997 Information Technology- telecommunications And Information exchange Between Systems-Local And Metropolitan Area Networks-specific Requirements-part 11: Wireless Lan Medium Access Control (MAC) And Physical Layer (PHY) Specifications.

[2] Bill McFarland and Michael Wong. The family dynamics of 802.11. *Queue. ACM Press*, 1(3):28–38, May 2003.

[3] Russ Housley and William Arbaugh. Security problems in 802.11-based networks. *Communications Of The ACM*, 46(5):31–34, May 2003.

[4] G. Racherla and D. Saha. Security and privacy issues in wireless and mobile computing. *IEEE International Conference on Personal Wireless Communications*, pages 509–513, 2000.

[5] Jon Allen and Jeff Wilson. Securing a wireless network. In *Proceedings of the 30th annual ACM SIGUCCS conference on User services*, pages 213–215. ACM Press, 2002.

[6] Bruce Schneier. *Applied Cryptography: Protocols, Algorithms and Source Code in C*, chapter 17, pages 397–398. John Wiley and Sons, Inc., second edition, 1996.

[7] Nikita Borisov, Ian Goldberg, and David Wagner. Intercepting mobile communications: the insecurity of 802.11. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 180–189. ACM Press, 2001.

[8] Scott Fluhrer, Itsik Mantin, and Adi Shamir. Weaknesses in the key scheduling algorithm of RC4. *Lecture Notes in Computer Science*, 2259:1–24, 2001.

[9] William A. Arbaugh, Narendar Shankar, and Y.C. Justin Wan. Your 802.11 wireless network has no clothes. *IEEE Wireless Communications*, 9(6):44–51, December 2002.

[10] Adam Stubblefield, John Ioannidis, and Aviel D. Rubin. A key recovery attack on the 802.11b wired equivalent privacy protocol (wep). *ACM Trans. Inf. Syst. Secur.*, 7(2):319–332, 2004.

[11] J. Postel and J. Reynolds. A standard for the transmission of ip datagrams over ieee 802 networks. Technical Report RFC 1042, IETF - Network Working Group, February 1988.

[12] Jeremy Bruestle, Blake Hegerle, and Snax. Airsnort. http://airsnort.shmoo.com/.

[13] Richard Matthew Stallman. Gnu general public license, 1989. http://www.gnu.org/copyleft/gpl.txt.

[14] Jouni Malinen. Host ap driver for intersil prism2/2.5/3. http://hostap.epitest.fi/.

[15] Anton T. Rager. Wepcrack. http://wepcrack.sourceforge.net/.

[16] Securityfocus bugtraq mailing list. http://www.securityfocus.com/archive/1.

[17] Ieee p802.11 - task group i. http://grouper.ieee.org/groups/802/11/Reports/tgi_update.htm.

[18] Nancy Cam-Winget, Russ Housley, David Wagner, and Jesse Walker. Security flaws in 802.11 data link protocols. *Commun. ACM*, 46(5):35–39, 2003.

[19] Vebjorn Moen, Havard Raddum, and Kjell J. Hole. Weaknesses in the temporal key hash of wpa. *SIGMOBILE Mob. Comput. Commun. Rev.*, 8(2):76–83, 2004.

[20] IEEE. *IEEE Std 802.1X-2001*, 2001. IEEE standard for local and metropolitan area networks - Port-based network access control.

[21] National institute of standards and tecnology. Gaithersburg, Maryland, http://www.nist.gov/.

[22] Bruce Potter. Wireless security's future. *IEEE Security & Privacy Magazine*, 1(4):68–72, August 2003.

[23] C. Brian Grimm. Wi-fi protected access overview, October 2002. http://www.wi-fi.org/.

[24] David L. Pepyne, Yu-Chi Ho, and Quinghua Zheng. Spring: Synchronized random numbers for wireless security. *IEEE Wireless Communications and Networking*, 3(3):2027–2032, March 2003.

[25] N. Chandran and D. Sampath. Strengthening wep protocol for wireless networks using block chaining algorithm with variable encrypting function mechanism. *IEEE/Sarnoff Symposium on Advances in Wired and Wireless Communications*, (April):141–143, 2004.

[26] Clive Ka-Lun, Xiao-Hui Lin, and Yu-Kwong Kwok. A multipath ad hoc routing approach to combat wireless link insecurity. *IEEE International Conference on Communications*, 1:448–452, May 2003.

[27] Yasir Zahur and T. Andrew Yang. Wireless lan security and laboratory designs. *J. Comput. Small Coll.*, 19(3):44–60, 2004.