

# Domande per Orale

## come funziona 802.1x + 802.11i?

L'802.1x e 802.11i sono due protocolli che combinati permettono autenticazione e crittografia in *reti protette*.

Infatti, la sicurezza di una rete non può basarsi sulla crittografia e basta, ma deve avere anche una base di autenticazione: **bisogna che gli utenti che vogliono collegarsi alla WiFi siano autenticati**.

Parliamo prima di tutto del protocollo 802.1x che è un protocollo port based network access control, dove nessuno passa se non è autenticato.

Esso coinvolge tre attori:

- Authenticator: è l' AP e controlla il traffico del Supplicant fino a che non è autenticato. Inoltra i messaggi all' **Authentication Server**
- Supplicant: colui che chiede di accedere alla WiFi, esso contiene credenziali dell'utente(nome utente, password e certificato).
- **Authentication Server**: verifica le credenziali dell'utente, se autenticazione va a segno esso comunica all'access point di aprire le porte e invia le chiavi sessione.

Questo colloquio si ha attraverso un tipo di messaggi a livello 2 (Collegamento) EAPoL facente parte del protocollo EAP.

Quello che fa è:

1. Access Point fa un EAP Request quando Supplicant si collega all'access point.(unicast). Il messaggio è incapsulato in un frame EAPoL e lo manda al cliente con il corrispettivo MAC address.
2. A questo punto il Supplicant rimanda un messaggio incapsulato EAPoL verso l'AP.
3. L'AP incapsula il messaggio EAPoL (lato wireless) in un messaggio RADIUS( lato ethernet) e mandato al Server Autenticatore.
4. L'Autenticatore autentifica in base alla tipologia di EAP e restituisce l'accettazione dell'accesso e manda chiavi di sessione a AP e Supplicant.

La chiave di sessione serve a criptare i dati tra AP e Supplicant, viene chiamata PMK.

Tramite questa attraverso il protocollo 802.11i si creano chiavi effimere PTK attraverso la **4-Way Handshake**. Queste chiavi effimere sono diverse per ogni utente.

Con PTK -> trasmissioni unicast.

Con GTK -> trasmissioni multicast.

# Routing Information Protocol (RIP):

E' un protocollo di Routing per vedere miglior percorso di routing, ogni 30 secondi un nodo manda ai suoi vicini la sua tabella di routing che esplicita: "Ecco quanti hop ci vogliono per andare al nodo \* (max 15 hop)".

Difetti:

- non scala bene su reti grandi.
- non converge velocemente.
- possibili loop di routing.

Open Shortest Path First (OSPF):

Ogni router manda info dei suoi link (LSA - Link State Advertising), tutti i router calcolano il grafo della rete ed attraverso Djakstra calcolano il path più corto. Vengono aggiornate se c'è un cambiamento della rete.

# Diffie Hellman:

E' un algoritmo che permette di calcolarsi una chiave simmetrica, ovvero, una chiave segreta comune solo a due utenti che vogliono comunicare.

1. Dati due utenti A e B, entrambi decidono due componenti principali in modo pubblico:
  - un numero primo grosso  $p$ .
  - un generatore  $g$  tale che  $1 < g < p - 1$
2. A questo punto A e B scelgono dei numeri segreti  $a$  e  $b$ . Questi sono privati e mai trasmessi.
3. Vengono calcolati due numeri adesso:

$$A = g^a \mod p$$

$$B = g^b \mod p$$

4. A e B si scambiano i numeri appena calcolati e rispettivamente calcolano il prodotto:

$$A \cdot B = g^{a \cdot b} \mod n = g^{b \cdot a} \mod n$$

E questa ha lo stesso valore sia per A che per B.

La sicurezza di questo algoritmo si ripone proprio sul logaritmo discreto, infatti questo algoritmo appoggia sulla definizione di funzione trappola.

Una funzione trappola è una funzione che è facile da calcolare in un modo ma è difficile calcolare l'inversa di questa. A meno che non si abbia una "porta segreta".

RSA

Questo tipo di algoritmo permette di calcolare chiavi asimmetriche, quindi una pubblica e una privata.

Funziona in questo modo:

1. Due numeri primi grandi  $p$  e  $q$
  2. Si calcola  $n = p \cdot q$  e si sceglie  $e$  compreso tra 1 e  $\phi(n) = (p - 1) \cdot (q - 1)$  tale che  $\gcd(e, n) = 1$
  3. Poi si sceglie  $d$  tale che  $e \cdot d = 1 \pmod{\phi(n)}$  con  $d$  che sta tra 0 ed  $n$
- La chiave pubblica è  $\{n, e\}$  mentre la chiave privata è  $\{n, d\}$ .

## NAT

Il NAT o Network Address Translation è uno strumento che permette di mappare un indirizzo pubblico con uno privato. In modo che i dispositivi di una rete si possono interfacciare al mondo del web attraverso un **singolo IP pubblico**.

Viene utilizzato principalmente per:

- conservare indirizzi IP pubblici
- aumentare la sicurezza poichè gli indirizzi privati non sono visibili su Internet, fornisce isolamento tra rete interna ed esterna.

I tipi di NAT sono:

- NAT statico: per ogni indirizzo privato ne abbiamo uno pubblico.
- NAT dinamico: gli indirizzi privati sono mappati ad UNO degli indirizzi pubblici disponibili in un pool di indirizzi.
- PAT (Porte): mappa gli indirizzi privati a uno pubblico differenziando le connessioni in base alla porta.

Vediamo il NAT con il protocollo UDP. L'UDP è un protocollo senza sessione e quindi senza stato. Quando un pacchetto UDP esce dalla rete il NAT affida quindi a quell'indirizzo privato quello pubblico a differenza di una porta.

Ma se viene usata la stessa porta per un altro dispositivo il NAT non è in grado di capire il destinatario per ambiguità causando un drop di pacchetti (non ideale per app in tempo reale).

È difficile monitorare anche il flusso perchè si ha un continuo cambio di Ip.

Queste accoppiate tra privato e pubblico si chiama binding.

I filtri del NAT possono decidere se fare la traduzione tra un binding o no, questo è dato proprio dalla caratteristica UDP che dicevamo (problema del demultiplexing).

NAT Simmetrico: tutte le richieste di un certo IP privato e porta e che hanno un IP di destinazione e porta sono mappati allo stesso IP pubblico e porta.

ecc...

## IPsec con NAT:

### IPSec in modalità transport o tunnel:

- Firma e/o cifra l'intero pacchetto.
- **Protegge anche gli header** (cioè gli indirizzi IP e le porte).
- Applica un **Integrity Check (HMAC)** per garantire che i pacchetti **non siano modificati** in transito.

Origine: 192.168.1.10 → Destinazione: 203.0.113.1

Contenuto: dati cifrati

Header IPSec: include HMAC

I NAT **modifica l'indirizzo IP sorgente** da 192.168.1.10 a 198.51.100.5, e magari anche la **porta** se usi UDP.

Il pacchetto cifrato è ancora:

Origine: 198.51.100.5 → Destinazione: 203.0.113.1

Contenuto: dati cifrati

**MA...** il problema è che:

- L'HMAC generato dal client A era basato su 192.168.1.10, **non** su 198.51.100.5.
- Il server riceve il pacchetto e fa il check dell'integrità → **✗ FALLISCE**, perché il pacchetto è stato *modificato dal NAT*.
- Risultato: **il pacchetto viene scartato**.

Si risolve con NAT T che fa:


- Il traffico IPSec viene **incapsulato in UDP** (porta **4500**).
- Questo permette al NAT di **modificare l'header UDP senza toccare il contenuto IPSec**.
- L'HMAC rimane valido perché la parte cifrata non viene toccata.

UDP Header (porta 4500)

|  
+--- IPsec (autenticato e cifrato)

|  
+--- Origine originale: 192.168.1.10

Il router NAT:

- Cambia 192.168.1.10 → 198.51.100.5
  - MA lo fa **solo nell'header UDP esterno**, non tocca IPsec
  - Il server spacchetta il pacchetto UDP, estrae il pacchetto IPsec e verifica l'HMAC → 
- PASSA**
- 

## DNS Poisoning

In questo attacco, si ha il DNS che sappiamo tradurre un URL di un sito in un IP, quando infatti si scrive nell'URL quello che si fa è mandare una richiesta DNS al server DNS che l'accoglie e guarda se ha il risultato nella cache.

Se non ce l'ha provvede a cercare in un altro server DNS autoritario.

Nel mentre che il DNS server aspetta una risposta, l'attaccante manda un messaggio che collega il dominio al suo indirizzo IP.

Per far sì che la sua risposta venga accettata:

- deve identificare **ID Transizione** che viene deciso dal client quando fa la richiesta al DNS.
- dominio richiesto
- la porta UDP della sorgente. (per richiesta di)

Se indovina allora il DNS accetta l'IP malevolo e così quando il cliente richiede quell'URL viene indirizzato in una pagina malevola.

## MITM - Man In The Middle

Attacco, nel quale, l'attaccante si posiziona nel mezzo a due che vogliono comunicare sottraendo informazioni e leggendo le conservazioni.

Attacco efficace, ad esempio, nel Diffie Hellman: poiché vengono mandati il generatore e la radice primitiva sulla rete in modo pubblico e chiaro.

Quindi l'attaccante può stabilire un collegamento rispettivamente con entrambi e funzionare come intermezzo leggendo, nel mentre, i loro dati.

O ad esempio nell' ARP Protocol, dove quando si manda nella rete locale l'ARP request e viene ricevuta al gateway dove è stato fatto un arp spoofing, allora l'ARP cache del gateway sarà avvelenato e restituirà il MAC dell'attaccante riuscendo, quindi, ad accedere ai dati della vittima.

## IDS - Intrusion Detection System

Questo sistema permette di individuare pacchetti malevoli nella rete in modo passivo, quindi, non scartando i pacchetti in questione come può fare il IPS (Intrusion Prevention System).

Abbiamo due tipi di IDS:

- **NIDS** (Network Intrusion Detection System): viene posizionato vicino ai gateway di rete per monitorare il traffico in entrata/uscita dalla rete o su uno switch SPAN/TAP.  
Dove SPAN/TAP sono switch che permettono di copiare il traffico, SPAN inoltra il traffico da una porta ad un'altra dove lo porta ad un IDS (congestione fa perdere pacchetti) mentre il TAP fa una copia esatta bit a bit e poi lo manda su una o più porte di monitoraggio non perde pacchetti ma è meno flessibile dello SPAN.
- **HIDS** (Host Intrusion Detection System): viene installato direttamente nell'host e controlla file e log del sistema.

Nel caso ci siano pacchetti malevoli, il sistema segnala all'amministratore della rete.

## Penetration Testing

Il penetration test viene utilizzato per testare la sicurezza di un sistema, organizzato da aziende per testare la propria sicurezza.

L'obiettivo è quello di trovare vulnerabilità, valutare impatto degli attacchi e fornire suggerimenti per correggere e mitigare i rischi.

E' strutturato in fasi:

- Ricognizione: porte, sistemi operativi del sistema, IP, DNS insomma recuperare informazioni sul target.
- Scanning: si utilizzano tool per scannerizzare porte aperte, servizi di ascolto.
- Gaining Access: tentativi di sfruttare vulnerabilità per accedere al sistema.
- Maintaining Access: tecniche per mantenere il controllo sul sistema.
- Covering Tracks: simulazione di come un attaccante si nasconderebbe nella realtà.
- Report Finale: report che analizza: tutte vulnerabilità, la loro gravità e il loro impatto, suggerimenti e migrazioni, prove e screenshot.

I tipi di Penetration test sono:

- Black Box: il tester non ha info utili per attaccare.
- Grey Box: il tester ha qualche info utile (ha accesso parziale / credenziali parziali).

- White Box: il tester ha tutto il codice.

## Risk Assignment e Risk Assessment(valutazione)

Per quanto riguarda il risk assignment, si parla di trovare, valutare e stimare dei potenziali rischi che potrebbero compromettere:

- confidenzialità
- integrità
- disponibilità

Include quindi individuazione minacce, analisi vulnerabilità e calcolo dell'impatto.

Si hanno due tipi di analisi:

- Qualitativa: si prende degli esperti e si assegna un valore( basso, medio, alto). (soggettiva)
- Quantitativa: si usano valori numerici e probabilità (costo, numero occorrenze). (oggettiva)
  1. Si da un valore economico ai dati in base alla loro importanza. (AV Asset Value)
  2. Si calcola il numero di occorrenze che si verifichi un certo rischio in un anno (ARO)
  3. Exposure Factor: rappresenta in percentuale che si verifichi su un assets
  4. Single Loss Expectancy (SLE):  $AV \cdot EF$
  5. ALE:  $SLE \cdot ARO$

Mentre il Risk Assessment, oltre a valutare i rischi e analizzarli, decide che azioni intraprendere. Si hanno queste fasi:

- Identificazione: si trovano le possibili vulnerabilità
- Analisi del rischio: se ne calcolano i costi in caso di avvenuta della vulnerabilità e gli si affida anche una probabilità.
- Trattamento/Valutazione: si decide che fare: si mitiga la vulnerabilità , si elimina , la accettiamo?

### MATRICI DEL RISCHIO

Per quanto riguarda il Risk Analysis, si assegna un asset ad un data e lui avrà un livello di sensibilità. Ad esso sarà collegato un valore economico e per questo avrà bisogno di un livello di protezione contro questi rischi. Ai rischi si daranno poi delle contromisure.

I Rischi vengono definiti da delle vulnerabilità del sistema e queste vengono quindi trovate grazie a dei threat agents che possono fare Penetrations Test o Code Review (per banchi).

---

## EAF Structure

EAF è un insieme di pratiche per costruire un architettura aziendale, esso aiuta a descrivere un sistema complesso dividendolo in domini, livelli o viste per facilitare la comprensione e il miglioramento continuo.

Ci sono vari tipi di EAF:

- Civile
- Militare
- Opensource

Le più importanti fasi sono:

1. Fase Preliminare: si definiscono idee e obiettivi.
2. Visione Architettura: si chiariscono scopi, idee.
3. Architettura del Business: definisce investimenti e profitti.
4. Requisiti: raccoglie e gestisce requisiti
5. Information System Architecture: specifica dati e tecnologie necessarie. Assume gli assets e da qui si definiscono quindi i requisiti di sicurezza e quindi un certo livello di protezione per ogni assets.

## Proprietà delle funzioni hash:

Le funzioni hash sono delle funzioni che prendono come input un certo valore e ne restituisce un altro.

Essa, però, ha le seguenti proprietà:

- Compressione:  $H: X \rightarrow Y$  con  $\dim(X) \ll \dim(Y)$
- Nessuna Collisione: ovvero che ogni input diverso deve avere output diverso e quindi non esistono due input che danno lo stesso output.
- Deve essere facile da calcolare
- E' difficile, a partire da un valore hashato, ricondursi all'input (Pre Image Resistance)
- Second Pre Image Resistance: dato un valore hash  $h$  e il suo input  $m$  è difficile trovare un altro  $m'$  tale che  $H(m)=H(m')$

## Network Management System

È un insieme di componenti che servono a monitorare, gestire, ottimizzare le reti informatiche. L'obiettivo principale è garantire che tutti i dispositivi e i servizi della rete funzionino correttamente, rilevare i problemi tempestivamente e supportare gli amministratori nella risoluzione e manutenzione.

Svolge le seguenti funzionalità :

- Fault Manegement: gestisce, quindi, errori, malfunzionamenti e disconnessioni.



- Performance Management: misura le prestazioni di rete.
- Configure Management: configura i dispositivi di rete.
- Accounting Management: rilevazione dell'uso della rete da parte di alcuni dispositivi.
- Security Management: monitoraggio di attività sospette (IDS,IPS)

L'architettura è fatta da:

- NMS centrale:
- Dispositivi gestiti
- SNMP -> protocollo di comunicazione tra i dispositivi della rete, è formato dai seguenti componenti:
  - SNMP manager: chiede info e risorse ai dispositivi della rete.
  - SNMP Agent: è un software installato nei dispositivi che manda info richieste dal manager
  - MIB: base generica per i dati che associa agents alle info reperibili

Il manager SNMP può richiedere le info attraverso il polling (richiedere ogni tot) o trap dove sono i dispositivi a mandare i loro dati.

## Web of Trust

Si basa sul fidarsi di chi gli altri si fidano, noi generiamo le nostre chiavi pubbliche e private, e come si sa, si distribuisce la chiave pubblica che viene firmata da quelli che sanno chi tu sia. Io stesso posso firmare la firma di qualcun altro se lo conosco.

Il problema, sta in un gruppo grande, dove qualcuno può iniettare qualcosa nel sistema che viene considerato vero dalle altre persone.

Ad esempio:

1. Un attaccante crea una chiave PGP falsa a nome di una persona nota, ad esempio:

 [linus@torvalds.org](mailto:linus@torvalds.org)



Genera una chiave pubblica che dichiara di essere Linus Torvalds (creatore di Linux).

2. Pubblica questa chiave **falsa** su un **keyserver pubblico** (che non richiede verifica).
3. Firma questa chiave **con altri account falsi** oppure riesce a farla firmare da utenti che:
  - **Non controllano bene** l'identità.
  - **Accettano automaticamente** chiavi per abitudine o pigrizia.
  - Sono **ingannati da nomi o indirizzi email simili** (es. [linus@t0rvalds.org](mailto:linus@t0rvalds.org)).
4. Tu, utente in buona fede, ricevi un messaggio da "Linus", controlli la chiave su un keyserver, e il tuo software ti dice:

✓ “Questa chiave è firmata da 3 persone in cui ti fidi. Tutto ok!”

MA potrebbe essere un malware.

## Certificati

Un certificato è qualcosa che non viene segnato dagli altri peers, ma da una Certificate Authority. Abbiamo, quindi, una informazione firmata da un'entità di cui si fidano tutti che ha verificato l'autenticità del documento.

Il certificato ha un tot di versioni ma lo standard è la seconda. Esso è formato da un tot di campi:

- Versione: versione del certificato.
- Numero Seriale: numero del certificato -> è distinto tra le diverse autorità di certificazione.
- ID dell'algoritmo per la firma: che algoritmo utilizzato
- Issuer X.500 name: nome dell'ente certificatore.
- Periodo di Validità: data di scadenza
- Subject X.500 name: è l'**identità dichiarata** dell'entità a cui è stato rilasciato il certificato. Serve affinché chi riceve il certificato possa sapere **a chi appartiene** e confrontare quell'informazione con ciò che si aspetta (ad esempio, il nome di dominio del sito web visitato).
- Subject public key info
- Extension: info extra
- Subject Unique ID: levare ambiguità tra i soggetti.
- Issuer Unique ID: levare ambiguità tra gli issuers
- CA Digital Signature: questa firma è per dire che il certificato è vero. Viene creato dall'hash del certificato stesso e criptato con la chiave privata della CA.

Il client genera chiavi asimmetriche (tramite RSA)  $k_{priv}$  e  $k_{pub}$ . A quel punto richiede un certificato (CSR) mandando info e chiave pubblica. Il CA riceve il  $k_{pub}$  e le info e firma digitalmente con la sua **chiave privata** "firma = Sign(identità ||  $k_{pub}$ , CA\_priv)".

Quindi costruisce il certificato e viene messa la firma nel campo CA Digital Signature e invia C al cliente. Il cliente verifica la firma della CA Verifica(identità ||  $k_{pub}$ , firma, CA\_pub), fa l'hash di C e poi confronta i due risultati.

### Autenticazione del server (dimostrazione del possesso della chiave privata)

Ora il browser chiede al server di **dimostrare che possiede la  $k_{priv}$**  associata alla  $k_{pub}$  del certificato:

- Tipicamente, il browser cifra una **PreMaster Secret** con  $k_{pub}$

- Il server la **decifra con la sua k\_priv**

Da lì entrambi derivano la chiave di sessione simmetrica 

Se tutto funziona → il browser sa che:

- Il certificato è **valido**
  - Il server è **realmente in possesso della chiave privata**
- E fanno una trasmissione criptata con questa chiave simmetrica.
- 

Ma come facciamo a sapere che la chiave pubblica della CA è effettivamente sua?

Sostanzialmente abbiamo nel sistema operativo un insieme di chiavi pubbliche delle CA che vengono utilizzate per autenticare siti web tramite certificato.

CA ferma le sue operazioni -> Certificate Revocation List CRL.

Se una CA è stata compromessa e la sua chiave è stata leakata -> i browser smettono di accettare i certificati in questione.

TLS HTTPS

## Curva Ellittica

## Firewall

Il firewall è un componente importante per la sicurezza della rete, dove quello che viene fatto è controllare tutti i pacchetti del traffico. Se, però, viene bypassato è inutile.

Deve essere ridondante e efficiente.

Può essere sia hardware che software. La differenza tra hardware e software è che la seconda deve far sì che OS e i processi annessi siano bug free, tale che non compromettano la sicurezza del firewall.

Il firewall va posizionato tra una parte sicura (rete aziendale) e una parte non sicura (rete internet).

Ci sono vari tipi di Firewall:

- Firewall che filtra pacchetti: ispeziona i pacchetti e vede se corrispondono alle regole per far passare.
- Firewall stateful: il firewall ha una memoria e decide sui pacchetti guardando cosa è successo nel passato ai livelli 3 e 4.

- Firewall del livello applicazione: fa un controllo dei dati che sono trasmessi (viola abbastanza la privacy).

Dove si mette il Firewall:

- rete interna: dove stanno i dispositivi utenti, i server interni e il database.
- zona Demilitarizzata: zone che contengono indirizzi web, email e i server DNS (servizi che si interfacciano direttamente con la rete). Essa è una zona a cui può accedere internet e che non c'è grossi problemi nel caso di compromissione poichè si può reimpostare tutto.

Per sicurezza, conviene mettere due firewall per rendere attacchi più difficili.

## Netfilter (iptables)

È framework del kernel Linux e offre funzioni come NAT, filtraggio pacchetti e traduzione porte. Questo permette quindi di proibire ad alcuni pacchetti di non arrivare in alcune zone della rete. Il Netfilter è formato da due componenti importanti:

- iptables: permette all'amministratore del sistema di configurare le regole del filtraggio dei pacchetti
- l'altra componente è il kernel di questo software.

Un esempio di regola fatta da iptables:

```
iptables -t filter -D INPUT -dport 80 -j ACCEPT
```

Questa regola può essere spiegata come: "Accetta tutti i pacchetti che passano dalla porta 80". Iptables contengono più tabelle, che contengono più chain che contengono regole (definite per i pacchetti).

Le tabelle sono gruppi di catene per ogni scopo specifico.

**|Tabella|Scopo principale|**

|---|---|

|filter|**Controllo accessi** (default per firewall)|

|nat|**Modifica indirizzi** (es. NAT, DNAT, SNAT)|

|mangle|**Manipolazione pacchetti** (TTL, TOS, QoS...)|

|raw|Bypass conntrack (più basso livello)|

|security|Integrazione con moduli SELinux/AppArmor|

Ogni riga della tabella contiene, una catena che rappresenta un insieme di regole.

Tabella: filter

|

|— Chain: INPUT

- | └─ Regola 1: accetta SSH da IP X
- | └─ Regola 2: accetta ICMP (ping)
- | └─ Regola 3: blocca tutto il resto
- |
- | └─ Chain: OUTPUT
- | └─ Regole per pacchetti generati dal sistema
- |
- | └─ Chain: FORWARD
- | └─ Regole per pacchetti che transitano nel sistema (es. router)

L' iptable ha 4 tabelle precostruite e ognuno di queste ha delle catene e delle regole:

- filter table: usata per decidere se accettare, scartare o bloccare un pacchetto

Catena	Descrizione
INPUT	Per i pacchetti <b>che arrivano al sistema</b> (es. pacchetti verso il tuo server SSH)
OUTPUT	Per i pacchetti <b>generati dal sistema stesso</b> (es. un ping che parte dal tuo computer)
FORWARD	Per pacchetti <b>che passano attraverso il sistema</b> (es. il tuo PC è un router o bridge)

Le azioni da eseguire, **target**, sono: drop,reject, accept,log

- NAT table: Usata per modificare **indirizzo IP e/o porta** nei pacchetti. (DNAT e SNAT)
- mangle table: manipolazione intestazione IP dei pacchetti o marcarli. Fatto dopo il CONNTRACK.
- raw table: **Disattivare il tracking** dei pacchetti. Fatto dopo il conntrack

CONNTRACK è una feature che attacca un tag ad ogni pacchetto, in un campo che può essere letto da chiunque cosicchè tiene d'occhio certi pacchetti.

## Firewall backup

Si possono avere due firewall per essere più sicuri affinché se se ne guasta uno può essere sostituito da un altro.

Ma il secondo non può essere completamente spento ma deve essere pronto con una configurazione automatica chiamata **heartbeat**.

Se la CPU della macchina va a zero per qualche ragione o è congestionata al 100% per troppo tempo, il cuore smette di pompare e il monitor capisce che il device ha fallito svegliando il

secondo device.

Ci sono tre tipi di backup primari:

- cold swap backup: la CPU non ha più potenza e l'heartbeat decide di accendere il secondo firewall ma deve comunque accendersi e avere una fase di boot.
- hot swap backup: il tempo di boot è più basso ma consuma di più poichè la memoria del secondo firewall viene alimentata ed è quindi attiva.
- active: il firewall di backup è attivo e funziona, gli switch gestiranno il traffico.

## WEP - WPA - WPA2

Il WEP è un algoritmo di sicurezza per le reti WiFi, il suo obiettivo è quello di creare la privacy tra i nostri dispositivi e l'AP.

Qui, viene introdotto il concetto di autenticazione e associazione dove il primo rappresenta il fatto che il dispositivo è in grado di entrare nella rete mentre il secondo ci dice se è nella rete. Il primo dice "So chi sei" mentre il secondo "puoi fare questo e quello".

L'autenticazione deve far dimostrare che sa qualcosa all' AP. Viene eseguito in 3 passi:

1. Il dispositivo manda una richiesta di autenticazione.
2. L'AP gli manda una challenge.
3. Il dispositivo risolve la challenge e gli risponde con la challenge criptata.

In questo caso la challenge criptata è il segreto e può rappresentare la password della rete.

L'associazione deve controllare di avere le risorse a disposizione (IP, ecc...)

Il **problema** di questo meccanismo per la challenge è che i nostri attacker possono vedere il plaintext mandato, di conseguenza , possono criptarlo (**information leakage**).

Attacchi Denial of Service (DoS):

- de associazione: l'attaccante si finge l'AP e fa rimandare le richieste di associazione alla vittima mentre l'AP autentico va in confusione perchè per lui è stato già autenticato l'utente.
- De Autenticazione IP: la vittima è obbligata a ripetere il processo di autenticazione, mandando il plaintext criptato del segreto di nuovo, utilizzando la criptoanalisi ci si può ricondurre al segreto.

## WEP

Parlando del cifrare i messaggi, il cifraggio è simile al One Time Pad, infatti viene fatto uno XOR bit a bit tra un segreto e il plaintext. Ora, il segreto non si intende quello calcolato a partire dalla password ma è calcolato tramite PRNG che è un algoritmo che prende come input un

seed e ne restituisce una stringa lunga che funge da chiave.

Ma come ogni generatore di numeri casuale, non si hanno veramente numeri casuali ma sarà una macchina stati che **prima poi ripartirà da capo** dopo una certa lunghezza chiamata *lunghezza di ciclo*. Di conseguenza si avrà sempre qualche correlazione.

Parlando del criptaggio WEP del payload, sappiamo che la chiave segreta è l'hash della password dello WiFi, quindi il segreto è comune a tutti e non cambia mai (a meno che non venga cambiata), ma comunque se viene scoperta l'attaccante può decryptare tutto il traffico di tutti gli utenti.

Il criptaggio del payload:

1. Hash della password della wifi per prendere la chiave segreta.
2. Si concatena IV (dimensione fissata, piccola 24 bit) e si mette al WEP PRNG per generare un flusso di byte randomico.
3. si concatena il plaintext al ICV (Integrity Check e quindi hash del plaintext).
4. Viene fatta la concatenazione XOR bit a bit tra il passo 2 e 3.
5. Viene aggiunto IV all'inizio del messaggio.

IV -> 24 bits (troppo corto si ripeterà dopo un pò di tempo e il secret è fisso quindi -,- )

Segreto -> 40 bits (troppo corto - si può calcolare calcolando 1 byte per parte di chiave)

PRNG -> RC4 (non è forte crittograficamente)

ICV -> CRC32 (buono ma è un semplice integrity check)

Attacchi a questo WEP:

- CHOP CHOP ATTACK: un attacco che ci permette di decryptare messaggio senza conoscere il segreto.
- blind attack: si cambiano dei bit, ad alcuni campi, anche se sono criptati se ci si riferisce a dei campi da bit singoli e gli si cambia bit assumerà il senso contrario.

## CHOP CHOP ATTACK

Come dicevamo prima il Chop Chop Attack è un attacco che permette di scoprire la chiave con il quale è stato criptato il messaggio senza saperne il segreto.

Abbiamo catturato un pacchetto con dei dati random dentro.

Si allega la figura:

Data						CRC			
$D_0$	$D_1$	$D_2$	$D_3$	$D_4$	$D_5$	$J_3$	$J_2$	$J_1$	$J_0$
$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$
$K_0$	$K_1$	$K_2$	$K_3$	$K_4$	$K_5$	$K_6$	$K_7$	$K_8$	$K_9$
$S_0$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$	$S_6$	$S_7$	$S_8$	$S_9$

TABLE 10.1: KoreK's chop-chop attack, first step.

Vediamo, quindi, da  $D_0$  a  $D_5$  sono i byte del pacchetto (ogni  $D$  è un byte) e da  $J_3$  a  $J_0$  si ha il CRC. Il risultato del criptaggio è segnato da  $S_0$  a  $S_9$ . Quello che si fa a questo punto è tagliare il pacchetto catturato con un byte in meno (si leva il byte relativo a  $D_5$ ).

DATA					CRC			
$D_0$	$D_1$	$D_2$	$D_3$	$D_4$	$I_3$	$I_2$	$I_1$	$I_0$
$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$
$K_0$	$K_1$	$K_2$	$K_3$	$K_4$	$K_5$	$K_6$	$K_7$	$K_8$
$S_0$	$S_1$	$S_2$	$S_3$	$S_4$	$R_5$	$R_6$	$R_7$	$R_8$

TABLE 10.2: KoreK's chop-chop attack, second step.

Qui, vedremo sicuramente, che la parte del CRC sarà sicuramente diversa dal primo pacchetto e ipotizziamo che la chiave sarà sempre la solita da  $K_0$  a  $K_9$ , taglieremo quindi la chiave  $K_9$ . Abbiamo detto che da  $S_0$  a  $S_4$  ci saranno gli stessi valori, ma da  $R_5$  ad  $R_8$  sarà diverso da  $S_5$  a  $S_9$ .

Questi valori dipenderanno esclusivamente da un valore tra 0 e 255 dettato da  $D_5$ , quindi, cambieremo quel byte finche non troveremo il CRC corrispondente. Quindi troveremo  $D_5$ , per trovare  $K_5$  basterà fare lo XOR tra  $D_5$  e  $S_5$ .

Come si fa a capire se va bene effettivamente? Non tramite il CRC ma mandando il pacchetto con il numero trovato e se è giusto riceveremo un ACK dal AP. Si fa per le altre parti nello stesso modo.

La cosa grossa è che dopo che si è trovato il keystream di un IV, riesco a calcolarmi il keystream di tutti gli altri IV. Mando un messaggio echo ICMP codificato con quell'IV. L'AP manderà un reply codificato con un keystream inerente al IV differente.



Creo un pacchetto reply che mi aspetto (facile da intuire) in chiaro e faccio lo XOR con quel reply che mi arriva.

Mi creo tantissime keystream inerenti ad ogni IV.

## WPA

L'RC4 non è male ma bisogna integrarlo in un sistema migliore , infatti, il PRNG non è più dipendente solo da IV e segreto ma da una chiave più lunga (128 bits) e molti più dati (TSC0,1,2,3,4 ; Trasmitt Address ; Temporary Key).

Questo nuovo protocollo si chiama TKIP.

Dopo questa parte si ha la stessa struttura della WEP, infatti, si concatena questo risultato a TSC0,1 e si passa ad il cifrario RC4 e poi viene fatto lo XOR.

Mentre il WPA2 ha cambiato il TKIP con AES e CCMP.

In questo caso nonostante sia stato rotto da KRACK, nel 2018, ha migliorie rispetto a WEP poichè, ad esempio, viene creata una chiave segreta temporanea e non una che è comune a tutti.

Questo permette che se la chiave temporanea finisce di un utente, finisce nelle mani di un attaccante quello che si ha è che lui non riuscirà a leggere tutto il traffico di tutti gli utenti ma solo dell'utente in questione.

## Tipo di Vulnerabilità

Ci sono diverse tipologie di vulnerabilità:

- vulnerabilità di design: viene progettato in questo modo, con la vulnerabilità che è conosciuta e ben accetta.
- vulnerabilità di bad design: dove si ha avuto una dimenticanza o un errore nella fase di design del software.
- vulnerabilità di bad implementazione: software non testato, programmatori sotto pagati, degli if incompleti, input non controllati. Bug fatali.
- vulnerabilità al bad deployment: il sistema lavora in laboratorio ma non nel campo.
- vulnerabilità di libreria: se il sistema si affida a librerie di terze parti con grosse vulnerabilità, il nostro sistema erediterà quelle.