

Indice generale

Introduzione.....	3
Capitolo 1 Sicurezza.....	6
1.1 Introduzione.....	6
1.2 Tipi di sicurezza	7
1.2.1 Caratteristiche di sicurezza.....	8
1.3 Attacchi.....	10
1.4 Sicurezza strato link.....	17
1.4.1 Attenuazione dei rischi di sicurezza della LAN.....	19
1.5 Sicurezza strato network.....	21
Capitolo 2 Framework AAA.....	24
2.1 Introduzione.....	24
2.2 Architettura AAA generica.....	26
2.3 Autorizzazione AAA.....	28
2.3.1 Single Domain.....	30
2.3.2 Roaming.....	31
2.3.3 Distributed Services.....	33
Fig 2.5 Distributed Services.....	33
2.4 Policies per l'autorizzazione.....	33
3.5 Autenticazione AAA.....	35
3.6 Accounting AAA.....	35
3.6.1 Analisi dell'andamento e pianificazione di capacità.....	36
3.6.2 Conto.....	36
3.6.3 Auditing.....	37
3.7 Scalabilità e affidabilità dell'accounting.....	37
Capitolo 3 RADIUS	38
3.1 Introduzione.....	38
3.2 Stabilire una sessione RADIUS.....	39
3.3 Caratteristiche del protocollo RADIUS.....	41
3.4 Metodi di autenticazione.....	43
3.4.1 PAP e CHAP.....	43
3.4.4 802.1x.....	47
Capitolo 4 RADIUS: Pacchetti e Attributi.....	51
4.1 Formato dei pacchetti.....	51
4.2 Tipi di pacchetti.....	54
4.3 Attributi.....	60
4.5 Sicurezza Radius.....	66
Capitolo 5 FreeRadius.....	70
5.1 Introduzione.....	70
5.2 Installazione FreeRadius.....	71

5.3 File di configurazione di FreeRadius.....	72
5.3.1 radiusd.conf.....	72
5.3.2 File di configurazione clients.conf.....	74
5.3.3 Il file di configurazione users.....	75
Appendice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point	76
A.1 Configurazione utenti.....	78
A.2 Configurazione di freeradius.....	80
A.3 LOG di connessione all'Access Point con autenticazione MSCHAPv2.....	92
Appendice B Esempio autenticazione captive portal: m0n0wall.....	95
B.1 Configurazione di m0n0wall.....	96
B.2 Configurazione Utenti.....	100
B.3 Configurazione di freeradius.....	102
B.4 File di log dell'autenticazione attraverso il captive portal.....	103
APPENDICE 3 file di configurazione di m0n0wall.....	105
Bibliografia.....	115

Ringraziamenti

Ringraziamenti

Vorrei ringraziare, innanzitutto, la mia famiglia che mi ha sostenuto in questo periodo un po' particolare.

Un ringraziamento al Dott. Fausto Marcantoni, per la sua pazienza e disponibilità.

Tutti gli amici, che non cito uno per uno perché siete veramente tanti, che mi hanno spronato in tutti i modi ad avvicinarmi a questo traguardo.

Introduzione

Introduzione

La Wireless LAN rappresenta una delle tecnologie più diffuse per le comunicazioni private, e il suo utilizzo nel mondo è in rapida crescita. Tuttavia se nelle reti wired il binomio trasmissione dati ed accesso alla rete è stato reso fisicamente sicuro e facilmente controllabile nelle reti wireless, invece, questa esigenza per molto tempo non ha trovato una soluzione concreta e solo con la pubblicazione nel Luglio 2004 dello Standard IEEE 802.11i si è posta in essere la piattaforma che meglio di tutte risolve le problematiche sulla sicurezza quali l'Autenticazione Client/Server, la Protezione Dati, il Controllo di Integrità ed il Non Ripudio.

Dal punto di vista pratico, il protocollo 802.1x non è altro che il frame di autenticazione EAP (Extensible Authentication Protocol) che si utilizza per l'autenticazione dei collegamenti point to point adattato per viaggiare in trame Ethernet invece che nei pacchetti PPP. In virtù di ciò il protocollo 802.1x è anche noto come EAPoL (EAP over LAN).

Nel capitolo 1 si fa una panoramica sul concetto di sicurezza in generale, valutando i tipi di attacchi e analizzando la sicurezza ai livelli link e networking del modello ISO/OSI.

Nel capitolo 2 si analizza il framwork AAA.

Nei capitoli 3 e 4 si analizza il protocollo RADIUS, in particolare nel capitolo 3 la struttura generale e nel capitolo 4 i pacchetti RADIUS e i suoi attributi.

Il capitolo 5 mostra, invece, FreeRadius, il software GNU/GPL utilizzato per implementare il sistema.

Nell'appendice A viene implementato un sistema di autenticazione con freeradius come server RADIUS con autenticazione EAP/PEAP, mysql database per

Introduzione

memorizzare gli utenti e un Access Point

Nell'appendice B viene implementato il captive portal con l'uso del software m0n0wall.

Entrambi questi metodi hanno i loro pregi e difetti. Utilizzando RADIUS con 802.11i si ha una maggiore sicurezza dovuta al fatto che oltre all'autenticazione degli accessi, che avviene quando il client si associa all'access point, il layer 2 della comunicazione wireless è cifrato con delle chiavi crittografiche che vengono cambiate ad intervalli regolari. Dal canto suo, invece, un Captive Portal, si limita semplicemente ad autorizzare, attraverso mediante web login, gli accessi quando il client ha già ottenuto un indirizzo IP. In questo caso, se si pretende che i dati siano anche criptati si deve ricorrere ad altri espedienti come per esempio le VPN. Il captive portal ha il vantaggio di non richiedere alcuna configurazione lato client e di poter funzionare con qualsiasi hardware Wi-Fi. In realtà, dato che lavora a livello IP, il Captive Portal può proteggere gli accessi anche sulla rete cablata. Il captive Portal si adatta meglio nelle realtà come gli HotSpot in cui l'unico obiettivo è proteggere dall'accesso indiscriminato a Internet. Il sistema con server RADIUS, invece, oltre ad essere più complicato da configurare per l'utente, necessita del supporto da parte dell'hardware (access point e schede di rete wireless) e del sistema operativo. WPA e WPA2 con server RADIUS si adattano meglio in quelle situazioni in cui è indispensabile garantire la confidenzialità dei dati oltre all'autenticazione degli utenti.

Capitolo 1 Sicurezza

1.1 Introduzione

Per sicurezza si intende la salvaguardia dei sistemi informatici da potenziali rischi e violazioni dei dati. I principali aspetti che riguardano la protezione dei dati sono:

- la confidenzialità o segretezza: mantenere le informazioni fuori dalla portata degli utenti non autorizzati;
- l'integrità: la protezione dei dati nei confronti delle modifiche del contenuto effettuate da un intruso;
- la disponibilità: prevenzione alla non accessibilità, sia delle risorse che ai dati.

La protezione dagli attacchi informatici viene ottenuta agendo su più livelli: quello fisico e materiale e quello logico. Il primo livello si ottiene, ponendo, per esempio, i server in luoghi il più possibile sicuri, dotati di sorveglianza e/o di controllo degli accessi fisici. Il secondo livello prevede l'autenticazione e l'autorizzazione di un'entità che rappresenta l'utente nel sistema. Successivamente al processo di autenticazione, le operazioni effettuate dall'utente sono tracciate in file di log. Questo processo di monitoraggio delle attività è detto *accounting*.

Il livello di sicurezza viene implementato attraverso il framework AAA che verrà trattato nel capitolo 2.

Nessuna rete di computer è veramente sicura, ma come la sicurezza delle reti wireless si sovrappone a quella delle reti cablate tradizionali?

Capitolo 1 Sicurezza

Nelle reti di computer teoricamente è sempre possibile per un intruso osservare il traffico su ogni rete, ed è spesso possibile aggiungere o inserire anche del traffico non voluto.

Sia se si parla di reti wireless che di reti cablate la domanda reale è: sono abbastanza sicure?

Le reti wireless aggiungono un livello di complessità della sicurezza rispetto alle reti cablate. Le reti cablate inviano segnali elettrici o impulsi di luce attraverso i cavi, mentre le reti wireless propagano segnali radio attraverso l'aria e sono più facili da intercettare.

Naturalmente ogni azienda deve determinare per se stessa il livello di rischio che è disposta ad accettare quando implementa una rete, soprattutto per quando riguarda le reti wireless.

La sicurezza della rete, sia essa cablata che wireless, può essere minacciata facilmente, ad esempio interferendo con le normali attività mediante attacchi di tipo DoS, di cui parleremo più approfonditamente nella sezione 1.2, oppure forzando la rete e andando a modificare o prendere il controllo delle risorse messe a disposizione.

Comunque più una rete viene amministrata, maggiore diventa la necessità di sicurezza.

Resta il fatto che la rete sicura è quella che non è stata ancora sviluppata.

1.2 Tipi di sicurezza

Spesso si distinguono due concetti di sicurezza quella passiva e quella attiva.

Per sicurezza passiva si intende tutte quelle tecniche e strumenti di tipo *difensivo*, ossia quel complesso di soluzioni il cui obiettivo è quello di impedire che utenti

Capitolo 1 Sicurezza

non autorizzati possano accedere a risorse, sistemi, impianti, informazioni e dati di natura riservata. Il concetto di sicurezza passiva, pertanto, è molto generale: ad esempio, per l'accesso a locali protetti, l'utilizzo di porte blindate, congiuntamente all'impiego di sistemi di identificazione personale, sono da considerarsi componenti di sicurezza passiva.

Per sicurezza attiva si intende, invece, tutte quelle tecniche e strumenti mediante i quali le informazioni ed i dati di natura riservata sono resi intrinsecamente sicuri, proteggendo gli stessi sia dalla possibilità che un utente non autorizzato possa accedervi (confidenzialità) sia dalla possibilità che un utente non autorizzato possa modificarli (integrità).

Entrambi questi tipi di sicurezza sono tra loro complementari ed entrambe indispensabili per raggiungere il desiderato livello di sicurezza di un sistema.

Le possibili tecniche di attacco sono molteplici, perciò è necessario usare contemporaneamente diverse tecniche difensive per proteggere un sistema informatico, realizzando più barriere fra l'intruso e l'obiettivo.

Spesso l'obiettivo dell'intruso non è rappresentato dai sistemi informatici in sé, quanto piuttosto dai dati in essi contenuti. Quindi la sicurezza deve far in modo di impedire l'accesso ad utenti non autorizzati, ma controllare, anche soggetti con autorizzazione limitata riguardo alcune operazioni, per evitare che i dati appartenenti alla rete vengano copiati, modificati o cancellati.

Le violazioni possono essere molteplici: vi possono essere tentativi non autorizzati di accesso a zone riservate, furto di identità digitali¹ o di file riservati, utilizzo di risorse che l'utente non dovrebbe potere utilizzare.

¹ Insieme delle informazioni e delle risorse concesse da un sistema informatico ad un particolare utilizzatore del sistema stesso

1.2.1 Caratteristiche di sicurezza

La sicurezza è caratterizzata da due concetti fondamentali:

- sicurezza(security): una serie di accorgimenti atti ad eliminare la produzione di danni irreparabili all'interno del sistema;
- affidabilità(reliability): prevenzione da eventi che possono produrre danni di qualsiasi gravità al sistema.

La sicurezza si riferisce all'abilità del sistema di proteggere le informazioni e le risorse del sistema stesso nel rispetto della confidenzialità e dell'integrità. Un sistema sicuro non è vulnerabile ad attacchi che possono interferire con le informazioni captandole o modificandole in modo maligno.

L'affidabilità è l'abilità di un sistema di compiere le proprie funzioni a certe condizioni per un periodo di tempo specifico. Un sistema affidabile funziona in conformità ad alcuni requisiti predefiniti, e può compiere bene le funzioni che sono state stabilite durante il periodo di progettazione.

L'affidabilità ha a che fare con il funzionamento interno del sistema, mentre la sicurezza ha a che fare con gli attacchi esterni di possibili intrusi.

Se immaginiamo il sistema come una struttura circolare a due livelli, l'affidabilità è il nucleo, perché esso si occupa degli aspetti interni, il modo in cui opera un sistema, mentre la sicurezza è l'involucro, cioè racchiude il sistema nell'insieme e agisce come meccanismo di protezione contro attacchi esterni.

Tre cose devono essere messe in atto per rendere ogni ambiente di rete sicuro:

1. Il controllo degli accessi per limitare gli utenti che vogliono guadagnare l'accesso alla rete. Esso può avvenire attraverso diversi metodi di autenticazione i quali sono progettati per verificare che un utente sia chi dice di essere e guadagnare, così, i privilegi di rete. Una volta determinato

Capitolo 1 Sicurezza

che l'utente appartiene alla rete, avviene l'autorizzazione per determinare di quali servizi ha bisogno l'utente.

2. La privacy per nascondere le informazioni a chi non dovrebbe averle. Le trasmissioni sulla rete sono suscettibili di lettura se i pacchetti di dati non sono cifrati.
3. Autenticazione e integrità: l'autenticazione, in questo caso, verifica che il dispositivo (piuttosto che l'utente) sia legittimato e che i pacchetti di dati provengono dalla fonte che rivendicano e non sono stati vittima di un attacco di tipo spoofing da un dispositivo delle rete disonesto che usa le credenziali rubate. L'integrità assicura che i pacchetti non possano essere alterati durante il percorso anche se essi hanno origine da un dispositivo di rete legittimo.

1.3 Attacchi

Nella sicurezza informatica è ironicamente crudele che molte delle caratteristiche che rendono il computer semplice da usare o molto efficiente e i tool utilizzati per proteggere la rete possono essere usati per compromettere il computer e la rete stessa.

Le principali minacce sono:

- Port Scanning
- Sniffing
- Spoofing
- DoS
- Exploit

Capitolo 1 Sicurezza

- Man-in-the-middle

Normalmente quando un attacker porta avanti un attacco comprende sempre una combinazione delle varie tipologie di attacchi, in modo da mettere in crisi il sistema dell'utente colpito.

Diamo ora una breve descrizione di alcune principali tipologie di attacco.

Port scanning: questo tipo di attacco è simile ad un ladro che passa a controllare su ogni casa porte e finestre per vedere se ce n'è una aperta e quali sono quelle bloccate.

Questa tecnica, nella sua forma più basilare, invia una richiesta di connessione, soprattutto pacchetti UDP TCP e ICMP, ad un computer su ogni porta in modo sequenziale e prende nota di quali porte rispondono o sembrano aperte per una indagine più approfondita, verificando così i servizi di rete attivi su quel computer.

Di per se questa tecnica non è pericolosa, infatti, viene spesso utilizzata dagli amministratori di rete per effettuare controlli e manutenzione.

Dato che il port scanning rivela informazioni dettagliate che potrebbero essere usate da un eventuale attacker per preparare un tecnica mirata a minare la sicurezza del sistema, viene posta molta attenzione, da parte degli amministratori, a come e quando vengono effettuati port scanning verso i computer della rete.

Sniffing: all'interno di una determinata rete, username e password dell'utente sono generalmente trasmesse in chiaro, il che significa che le informazioni possono essere visibili analizzando i pacchetti che sono trasmessi.

L'attività di sniffing prevede l'intercettazione passiva dei dati che transitano in una

Capitolo 1 Sicurezza

rete . Tale attività può essere svolta sia per scopi legittimi, ad esempio l'individuazione di problemi di comunicazione o di tentativi di intrusione mantenendo la rete efficiente, sia per scopi illeciti come intercettazione fraudolenta di password o di altre informazioni sensibili.

I prodotti software utilizzati per eseguire queste attività vengono detti sniffer ed oltre ad intercettare e memorizzare il traffico offrono funzionalità di analisi del traffico stesso.

Gli sniffer intercettano i singoli pacchetti, decodificando le varie intestazioni di livello data link, rete, trasporto, applicativo. Inoltre possono offrire strumenti di analisi che analizzano ad esempio tutti i pacchetti di una connessione TCP per valutare il comportamento del protocollo o per ricostruire lo scambio di dati tra le applicazioni.

Per difendersi da questo tipo di attacco ci sono diversi metodi come la cifratura del traffico, in particolare delle informazioni sensibili, l'utilizzo di strumenti software in grado di rilevare la presenza di sniffer nella rete.

Spoofing: con il termine spoofing viene indicato un tipo di attacco utilizzato per falsificare l'identità del mittente del messaggio o dei file allegati. Si può così tentare di sfruttare l'identità falsa per muoversi nell'anonimato oppure per ottenere vantaggi o arrecare danni all'insaputa della vittima. Lo sniffing è un attacco passivo dove l'hacker si limita a monitorare i pacchetti che attraversano la rete, al contrario, lo spoofing è un processo attivo in cui l'hacker tenta di convincere un altro sistema che i messaggi da lui inviati provengono da un sistema legittimo. In altre parole, utilizzando lo spoofing, l'hacker simula di essere un altro utente o un altro sistema.

Le tecniche di spoofing sono diverse, le più note e adoperate sono:

Capitolo 1 Sicurezza

- **IP Spoofing:** E' il tipo di attacco più diffuso perché è il più facile da eseguire. L'attacco è reso possibile dal fatto che la maggior parte dei router attivi all'interno delle reti aziendali, controllano, durante la richiesta di accesso ai servizi, solo l'indirizzo IP di destinazione e non quello di provenienza. Questo tipo di attacco ha tanto più successo quanto più forti sono i rapporti di fiducia tra due o più macchine.

Gli attacchi Spoofing IP possono essere divisi in tre categorie:

- *IP Spoofing cieco:* quando l'attaccante cerca di farsi passare per un host di una qualsiasi sottorete.
 - *IP Spoofing non cieco:* è attuabile in una rete LAN quando chi attacca cerca di farsi passare per un host che è nella sua stessa sottorete.
 - *Attacchi dos:* l'attaccante cerca di bloccare un host per impedire a quest'ultimo di svolgere la normale attività oppure per prenderne il controllo.
- **Mail spoofing:** con questa tipologia di spoofing si fa apparire un allegato di una mail come se fosse di un tipo diverso da quello che è realmente. Questo attacco si basa su una vulnerabilità dei mime type usati per inviare e-mail. E' una tecnica semplice, i cui effetti possono essere disastrosi: basta modificare in maniera opportuna il nome dell'allegato da inviare.
esempio pratico :
 - Nome allegato: pippo.exe
 - Cambiamo il nome da pippo.exe a pippo.jpg

Quando l'e-mail arriva il client interpreterà il nome dell'allegato solo come pippo.jpg. l'ignaro utente cercherà di visualizzare il file, ma in realtà involontariamente eseguirà pippo.exe. L'esecuzione di un programma creato ad

Capitolo 1 Sicurezza

hoc può portare alla perdita di dati oppure all'apertura di backdoor sulla macchina vittima.

- *Web spoofing* Il web spoofing consiste nel far credere ad un utente che sta visitando il sito web desiderato, con la pagina richiesta mentre ne guarda una modificata. Questa tecnica fa uso massiccio di javascript. Supponiamo di visitare `www.pippo.net`. La pagina principale si frappone tra il nostro client e le pagine richieste successivamente. Si comporterà come un proxy non voluto o non visto, così potrà vedere tutto ciò che vede il client: siti, form, password.

Primo passo:

1. Il browser utente visita la pagina del web attaccante
2. Il browser utente decide di visitare un server fidato

Conseguenze

1. Invece di vedere direttamente la pagina del server fidato, il server web attaccante si connette al sito fidato e preleva la pagina richiesta dal browser client con javascript l'host nemico reindirizza la connessione
2. Modifica lo status del browser
3. Disabilita alcune funzioni dei menù del browser.

DoS: letteralmente Denial of service². In questo tipo di attacco si cerca di portare il funzionamento di un sistema informatico che fornisce un servizio, ad esempio un sito web, al limite delle prestazioni, lavorando su uno dei parametri d'ingresso, fino a renderlo non più in grado di erogare il servizio.

² Negazione del servizio

Capitolo 1 Sicurezza

Gli attacchi vengono attuati inviando molti pacchetti di richieste, di solito ad un server web, FTP o di posta elettronica saturandone le risorse e rendendo tale sistema instabile, quindi qualsiasi sistema collegato ad Internet e che fornisca servizi di rete basati sul TCP è soggetto al rischio di attacchi DoS.

Trattandosi di connessioni apparentemente legittime, è impossibile bloccarle senza interrompere anche il flusso di traffico inoffensivo. Però limitando drasticamente il numero di sessioni aperte simultaneamente l'impatto dell'attacco si riduce considerevolmente senza limitare il flusso dei pacchetti regolari.

Exploit: identifica un metodo che, sfruttando un bug o una vulnerabilità, porta all'acquisizione di privilegi.

Ci sono diversi modi per classificare gli exploit. Il più comune è una classificazione a seconda del modo in cui l'exploit contatta l'applicazione vulnerabile. Un *exploit remoto* è compiuto attraverso la rete e sfrutta la vulnerabilità senza precedenti accessi al sistema. Un *exploit locale* richiede un preventivo accesso al sistema e solitamente fa aumentare i privilegi dell'utente oltre a quelli impostati dall'amministratore.

Lo scopo di molti exploit è quello di acquisire i privilegi di root su un sistema. È comunque possibile usare exploit che dapprima acquisiscono un accesso con i minimi privilegi e che poi li alzano fino ad arrivare a root.

Normalmente un exploit può sfruttare solo una specifica falla e quando è pubblicato questa falla è riparata e l'exploit diventa obsoleto per le nuove versioni del programma. Per questo motivo alcuni hacker non divulgano gli exploit trovati ma li tengono riservati per loro o per la loro comunità. Questi exploit sono chiamati zero-day exploit, e scoprire il loro contenuto è il più grande desiderio per gli attacker senza conoscenze.

Capitolo 1 Sicurezza

Man-in-the-middle: questa tipologia di attacco consiste nel dirottare il traffico generato durante la comunicazione tra due host verso un terzo host, quello dell'intruso. Durante l'attacco si cerca di far credere ad entrambi gli end-point della comunicazione che il terzo host è in realtà il loro interlocutore legittimo.

L'host intruso riceve, quindi, tutto il traffico generato dagli end-point e si preoccupa di inoltrare correttamente il traffico verso l'effettiva destinazione dei pacchetti ricevuti.

Ci sono diverse forme che può assumere questo tipo di attacco a seconda dello scenario in cui opera.

Tra le più importanti ci sono:

- ARP poisoning
- DNS spoofing

Nell'ARP poisoning viene sfruttato il comportamento di un host in ricezione di una ARP reply che permette di modificare le ARP cache degli host vittime dell'attacco.

Si inviano falsi ARP reply ai due host che si vuole attaccare. Nelle risposte cambieremo l'indirizzo MAC di destinazione inserendo quello dell'host intruso.

Da quel momento in poi tutti i pacchetti, che dovrebbero viaggiare tra le vittime, in realtà, saranno spediti all'host intruso.

Nel DNS spoofing interessa sostituire l'indirizzo IP della risposta con quello dell'host intruso. Per far ciò è necessario creare delle risposte con il valore giusto del campo ID. Per scoprire l'ID basta utilizzare uno sniffer che intercetti i messaggi tra il client e il server DNS. Prima che il vero DNS risponda al client bisogna creare una risposta con l'ID sniffato ed inviarla al client.

Una volta fatto questo il client invierà tutti i pacchetti destinati al server DNS

Capitolo 1 Sicurezza

all'indirizzo IP dell'host intruso. L'intruso deve preoccuparsi di accettare la connessione e di crearne un'altra verso il server DNS reale.

1.4 Sicurezza strato link

Ogni livello della comunicazione ha le proprie sfide per riuscire a avere un buon livello di sicurezza. La comunicazione di livello Data Link³ è un collegamento debole in termini di sicurezza. La sicurezza della rete dovrebbe essere indirizzata su più livelli per combattere le diverse vulnerabilità.

Gli switch sono componenti importanti per la comunicazione di livello 2 e, inoltre, sono spesso usati per la comunicazione di livello 3. Essi sono suscettibili di molti attacchi subiti, al livello 3, dai router, oltre a quelli propri, come:

- CAM (Content-Addressable Memory) table overflow: la tabella CAM in uno switch contiene informazioni come gli indirizzi MAC disponibili su una porta fisica di uno switch, oltre ai parametri associati alle VLAN. La dimensione della tabella è limitata. Tipicamente un intruso sommergerà lo switch con tantissimi indirizzi MAC con la sorgente non valida fino a che la tabella CAM si riempie. Quando questo avviene lo switch sommergerà le porte con il traffico ricevuto perché esso non può trovare il numero della porta associato un particolare indirizzo MAC nella tabella CAM. L'overflow della tabella CAM fa affluire il traffico all'interno della rete VLAN così che l'intruso vedrà solamente il traffico all'interno della VLAN alla quale è collegato.
- VLAN hopping⁴: è un attacco dove un end system invia pacchetti destinati

³ Livello 2 del modello ISO/OSI

⁴ drogare

Capitolo 1 Sicurezza

ad un sistema su una VLAN differente che non può normalmente essere raggiunta da un end system. Questo traffico è taggato con un VLAN ID diverso da quello dell'end system a cui appartiene. Il sistema d'attacco può provare a comportarsi come uno switch e negoziare un canale in modo che l'attacker possa inviare e ricevere il traffico tra le altre VLAN.

- Manipolazioni dello Spanning-Tree Protocol: questo protocollo è usato nelle reti switch per impedire la creazione di loop di bridging in una rete Ethernet. Attaccando lo Spanning-Tree, l'attacker spera di eseguire un attacco di tipo spoof cercando di diventare il bridge di root nella topologia. Per fare questo l'attacker invia in broadcast i BPDU⁵ sul cambiamento della topologia della rete nel tentativo di rifar calcolare lo Spanning-Tree. I BPDU inviati dal sistema dell'attacker rivelano che il sistema d'attacco ha una bassa priorità di bridge. Se è andato a buon fine, l'attacker può vedere diversi frame.
- Spoofing dell'indirizzi MAC: questo tipo di attacco implica l'uso di un indirizzo MAC conosciuto di un altro host per tentare di far cambiare l'obiettivo dei frame inviati dallo switch all'host remoto verso la rete dell'attacker. Trasmettendo un singolo frame con l'indirizzo Ethernet sorgente di un altro host, l'attacker della rete sovrascrive la entry della tabella CAM così che lo switch, invece di inviare i pacchetti all'utente designato li invia all'attacker. Fino a che l'host non invia traffico esso non riceverà traffico. Quando l'host invia traffico, la entry della tabella CAM viene riscritta ancora una volta così che esso ritorna alla porta originale.
- Attacco ARP: l'ARP viene utilizzato per mappare gli indirizzi IP in indirizzi MAC in un segmento della LAN dove risiedono host della stessa sottorete. L'attacco ARP avviene quando qualcuno prova a cambiare la tabella ARP con le informazioni degli indirizzi MAC e IP senza

⁵ BPDU: Bridge Protocol Data Unit

Capitolo 1 Sicurezza

autorizzazione. Facendo così, gliacker possono compiere attacchi di spoofing sugli indirizzi MAC e IP per lanciare o attacchi di tipo DoS o di tipo Men-in-the-Middle.

- VLAN private: le VLAN private lavorano limitando le porte all'interno di una VLAN che possono comunicare con altre porte nella stessa VLAN. Le porte isolate all'interno della VLAN possono comunicare solo con le porte promiscue. Le porte di una comunità possono comunicare solo con altri membri della stessa comunità e con le porte promiscue. Le porte promiscue possono comunicare con ogni porta. Un attacco capace di bypassare la sicurezza della rete delle VLAN private coinvolge l'uso di un proxy per bypassare le restrizioni di accesso ad una VLAN privata.

1.4.1 Attenuazione dei rischi di sicurezza della LAN

L'attacco CAM table-overflow può essere attenuato configurando il port security sullo switch. Questa opzione fornisce le specifiche degli indirizzi MAC su una particolare porta dello switch o le specifiche del numero di indirizzi MAC che possono essere appresi da una porta dello switch. Quando un indirizzo MAC non valido viene individuato sulla porta, lo switch può o bloccarlo o mettere la porta nello stato down.

L'attenuazione dell'attacco VLAN hopping richiede diverse modifiche alla configurazione della VLAN. Uno degli elementi più importanti è l'uso di ID VLAN dedicati per tutte le trunk port. Inoltre, disabilitare le porte dello switch non utilizzate e metterle in una VLAN non utilizzata. Settare tutte le porte dell'utente nella modalità nontrunk specificando la chiusura di DTP su queste porte.

Capitolo 1 Sicurezza

Per attenuare la manipolazione del protocollo Spanning-Tree usare il root guard e il BPDU guard per far rispettare il collocamento del bridge di root nella rete così come far rispettare lo Spanning-Tree protocol domain border. La caratteristica root guard è progettata per fornire un modo per imporre il posto di root-bridge nella rete.

Lo Spanning-Tree protocol BPDU guard è progettato per permettere al progettista della rete di mantenere attiva la topologia della rete. Mentre il BPDU guard può sembrare non necessario poiché l'amministratore può settare la priorità bridge a zero, non è garantito che esso sarà eletto come bridge di root perché ci potrebbe essere un bridge con priorità zero e un bridge ID più basso. BPDU guard è meglio sviluppato verso le porte user-facing per prevenire un malintenzionato ampliamento della rete switch da parte di un attacker.

Usando i comandi port security per attenuare l'attacco MAC-spoofing. Questa opzione fornisce la capacità di specificare gli indirizzi MAC del sistema connesso ad una particolare porta. Il comando fornisce anche la capacità di specificare un'azione da intraprendere nel caso in cui avvenga una violazione della sicurezza della porta. Hold-down timer nel menù di configurazione delle interfacce può essere usato per diminuire la possibilità di un attacco di tipo ARP spoofing settando il periodo di tempo un entry rimarrà nella ARP cache.

Per attenuare la possibilità di un attacco private VLAN è possibile configurare delle ACL sul router.

In più lo standard IEEE 802.1x per far passare il framework EAP su reti cablate e reti wireless, agisce come portiere per l'accesso base alla rete al livello datalink. Negando l'accesso alla rete prima dell'autenticazione, l'802.1x può prevenire molti attacchi verso le infrastrutture di rete che dipendono dalla connettività.

Le misure di sicurezza al livello datalink sono complementari a quelle del livello network per fornire più protezione alla rete e agli utenti, specialmente nel caso di

reti wireless.

1.5 Sicurezza strato network

Il livello Network⁶ è vulnerabile in particolare per quanto riguarda gli attacchi di tipo DoS e problemi di sicurezza delle informazioni. Il protocollo più popolare usato nel livello network è l'IP.

Nel seguente paragrafo verranno analizzati i rischi chiave della sicurezza associati al protocollo IP.

- **IP Spoofing:** l'intruso invia messaggi all'host con un indirizzo IP indicante che il messaggio proviene da un host fidato per guadagnare un accesso non autorizzati all'host o ad un altro host. Per attuare un IP spoofing, un hacker deve prima usare diverse tecniche per trovare un indirizzo IP di un host di fiducia e poi modificare l'header del pacchetto così che possa sembrare che i pacchetti arrivano da quell'host.
- **Routing (RIP⁷) Attack:** il protocollo RIP è usato per distribuire le informazioni di routing, come il percorso più corto, e pubblicare le rotte fuori della LAN. RIP non è stato sviluppato per l'autenticazione, e le informazioni fornite dai pacchetti RIP sono spesso usate senza che si possano verificare. Un attacker potrebbe falsificare un pacchetto RIP, affermando che il suo host X ha il percorso più veloce nella rete. Tutti i pacchetti inviati sulla rete sarebbero diretti verso l'host X, dove essi potrebbero essere esaminati o modificati. Un attacker potrebbe usare RIP per impersonare effettivamente ogni host, determinando che tutto il traffico inviato a questo host sia inviato, invece, alla macchina

6 livello 3 del modello ISO/OSI

7 Routing Information Protocol

Capitolo 1 Sicurezza

dell'attacker.

- ICMP attack: il protocollo ICMP è usato dallo strato IP per trasmettere messaggi informativi one-way. Anche nell'ICMP non c'è autenticazione, il che porta ad attacchi usando l'ICMP che causa un DoS, o permettendo ad un attacker di intercettare i pacchetti. Attacchi DoS usa o i messaggi l'ICMP "Time Exceeded" o quelli di "Destination unreachable". Entrambi questi messaggi possono causare un immediata caduta della connessione ad un host. Un attacker può fare uso di questi semplici tipi di contraffazione e inviare ad un host o a diversi host che fanno parte della comunicazione. La loro connessione sarà rotta. Il messaggio ICMP "Redirect" è comunemente usato dai gateway quando un host ha erroneamente presupposto che la destinazione non è nella LAN. Se un attacker falsifica un messaggio ICMP "Redirect", esso può indurre un altro host a inviare pacchetti per una determinata connessione attraverso l'host dell'attacker.
- Ping flood (ICMP flood): il PING è uno dei più comuni usi del protocollo ICMP che invia un ICMP "Echo Request" all'host, e aspetta da questo host che rinvii un messaggio ICMP "Echo Reply". L'attacker semplicemente invia un gran numero di ICMP "Echo Request" alla vittima per mandare il suo sistema in crash o rallentarlo. Questo è un attacco semplice perché molte utility del ping supportano questa operazione, e l'hacker non ha bisogno di molte informazioni e conoscenze.
- Ping of Death Attack: un attacker invia un pacchetto ICMP Echo request che è molto più grande della massima dimensione del pacchetto IP della vittima. Dato che il pacchetto ICMP echo request ricevuto è più grande della dimensione normale del pacchetto IP, la vittima non può riassemblare i pacchetti. Il S.O. può, come conseguenza, essere mandato in

Capitolo 1 Sicurezza

crash o reboot.

- Teardrop Attack: un attacker usando il programma Teardrop per inviare frammenti IP che non possono propriamente essere riassemblati manipolando il valore offset del pacchetto possono causare un reboot o un blocco del sistema vittima.
- Packet sniffing: poiché la maggior parte delle applicazioni di rete distribuiscono i pacchetti in chiaro, un packet sniffer fornisce al suo utente informazioni significative e spesso sensibili, come l'account dell'utente e le password. Un packet sniffer può fornire ad un attacker le informazioni interrogando un database, oltre al nome account dell'utente e le password usate per accedere al database. Questo causerà seri problemi di privacy delle informazioni.

Nella maggior parte dei problemi della sicurezza della rete, non ci sono soluzioni ottimali per risolverli, tuttavia, ci sono molte tecnologie e soluzioni disponibili per attenuare i problemi di sicurezza sopra citati e monitorare la rete per ridurre i relativi danni quando viene portato a termine l'attacco. I problemi come il PNG flood possono essere effettivamente ridotti schierando i Firewall in posizioni critiche della rete per filtrare traffico non voluto da destinazioni pericolose. Utilizzando IPsec e VPN al livello network e usando l'autenticazione di sessione e dell'utente e le tecnologie di cifrature dei dati al livello datalink, il rischio di un IP spoofing e di un Packet sniffing sarà ridotto significativamente. Ipv6 in combinazione con IPsec forniscono migliori meccanismi di sicurezza per la comunicazione al livello network e ai livelli superiori.

Capitolo 2 Framework AAA

2.1 Introduzione

Il controllo degli accessi è uno degli elementi fondamentali per un'infrastruttura di rete, in questo modo è possibile controllare chi può avere accesso alla rete e cosa può fare, quali servizi può effettivamente utilizzare, dopo avere eseguito l'accesso. AAA è un framework attraverso il quale è possibile configurare il controllo degli accessi sugli apparati di rete, siano essi access-server, terminali mobili wifi. AAA è l'acronimo di Authentication, Authorization e Accounting. Questi sono le tre basi che si incontrano frequentemente in molti servizi di rete. Esempi di questi servizi sono l'accesso dial-in a Internet, commercio elettronico e Mobile IP.

Il framework AAA si propone di fornire una via modulare all'esecuzione dei seguenti servizi:

- Autenticazione: fornisce il servizio di identificazione degli utenti (risponde alla domanda chi è?) attraverso molteplici modalità: login/password, challenge/response, encryption.

L'autenticazione ci consente di identificare l'utente prima che sia autorizzato all'accesso alla rete o ai suoi servizi.

- Autorizzazione: fornisce il servizio per il controllo remoto degli accessi, autorizzazione per utente, autorizzazione per gruppo, autorizzazione per servizio, supporta IP, IPX, Telnet. Si basa sulla creazione di un insieme di attributi che descrivono le policy associate all'utente.

Un database locale o remoto contenente le informazioni per singolo utente servirà per comparare tale set di informazioni con quelle presenti al suo interno, il risultato sarà ritornato ad AAA che determinerà le azioni che

Capitolo 2 Framework AAA

l'utente può o non può compiere.

I server remoti, anche detti remote security server, autorizzano l'utente associandogli una coppia AV, attributo-valore, che definisce queste azioni.

- Accounting: fornisce il servizio di raccolta dati utilizzati per il billing, l'auditing, il reporting, analisi dell'andamento.

In questo modo possiamo tracciare le attività dell'utente, come i servizi utilizzati e le risorse di rete consumate.

Quando l'accounting è attivato l'access server invia dei report sull'attività utente al nostro security server, nel nostro specifico FreeRADIUS, sotto forma di accounting records, composto da una coppia AV di accounting che verrà salvata sul nostro server remoto, o access control server. Questo consente la possibilità di un'analisi a posteriori dei dati collezionati con scopi di auditing, billing e network management.

I principali protocolli AAA per amministrare le funzioni di sicurezza sono RADIUS[1] e DIAMETER[2]. Se il nostro router o access server ha funzionalità di NAS⁸, AAA rappresenta la modalità attraverso la quale stabiliamo la comunicazione tra il NAS e il server RADIUS.

In questa sede si parlerà più approfonditamente del protocollo RADIUS più precisamente nei capitoli 3 e 4.

Un' infrastruttura AAA consiste generalmente di un rete di server AAA che interagiscono con l'un l'altro usando un protocollo AAA. I server AAA autenticano gli utenti, si occupano delle richieste di autorizzazione, e raccoglie i dati di accounting. L'infrastruttura può anche includere i brokers⁹. Essi sono usati per accordare la fiducia tra le entità che non hanno un rapporto di fiducia l'un l'altro, i broker agiscono come terze parti fidate.

Per esempio, i server AAA di diverse aziende possono scambiare informazioni

⁸ Network Access Server

⁹ Agenti

Capitolo 2 Framework AAA

usando i broker come mediatori

2.2 Architettura AAA generica

L'architettura generica AAA viene descritta in [3]. L'approccio adottato in questa architettura è di dividere le funzionalità del framework AAA in due parti: una parte generica, che definisce le funzionalità comuni a tutte le applicazioni, e una parte specifica dell'applicazione, che definisce le funzionalità di specifiche applicazioni.

L'autorizzazione di un utente, per consentirgli l'uso di certi servizi, può essere divisa in tre passi: una richiesta di autorizzazione, una decisione basata su una policy e una risposta o un'azione.

Un utente, richiedendo l'autorizzazione, pone la sua richiesta ad un server di autorizzazione. Il server riceve la richiesta e prende una decisione. Questa decisione è basata sulle policy che il server ha per questo utente o per il servizio che l'utente ha richiesto.

Il processo di autorizzazione nell'architettura AAA è diviso in tre passi. Il primo passo di questo processo è la richiesta. Quando un utente richiede l'autorizzazione, il rule based engine (RBE) del server AAA riceve la richiesta. Esso contatta un database chiamato policy repository. Questo database contiene tutte le policy per i servizi che questo server offre. Una policy può richiedere la verifica di una variabile esterna.

Il mondo interno del server AAA è basato su una logic, una variabile che può essere vera o falsa, dato che la risposta ad una richiesta di autorizzazione può essere affermativa o negativa.

Quando una policy richiede la comunicazione con dispositivi esterni per verificare

Capitolo 2 Framework AAA

una variabile esterna, c'è bisogno di un'interfaccia tra il mondo interno e la semantica fuori dal mondo. Questo collegamento è fornito da Application Specific Module (ASM).

Nel verificare una variabile esterna, il RBE contatta il giusto ASM. L'ASM, allora, comunica con i dispositivi esterni per determinare la variabile esterna usando il protocollo specifico per quel dispositivo. Questa variabile viene poi inviata all'ASM che poi invia un valore logico al RBE. Il RBE alla fine decide se concedere o meno l'autorizzazione richiesta all'utente.

Oltre ad occuparsi della verifica delle variabili interne per l'assegnazione delle richieste di autorizzazione, gli ASM si occupano, anche, di mettersi in contatto con i servizi richiesti. Per esempio, quando un utente è autorizzato per l'uso della bandwidth di un router un ASM contatterà il router per “dirgli” di riservare la bandwidth richiesta per l'indirizzo IP dell'utente.

Nella figura 2.1 viene mostrato uno schema semplificato dei componenti interni di un server AAA. Essa mostra la connessione tra il RBE, le policy repository e gli ASM.

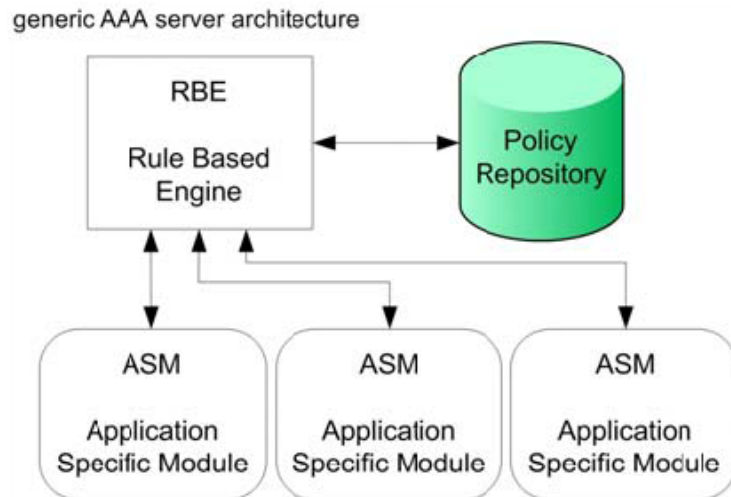


Figura 2.1 componenti interni di un server AAA

2.3 Autorizzazione AAA

Il documento [4], dell'AAAarch Research Group di Internet Engineering Task Force (IETF), fornisce un insieme di patterns architetturali per la realizzazione dei servizi di Authentication, Authorization and Accounting (appunto AAA) distribuiti.

Il modello di controllo di accesso previsto in [4] è quello in cui un utente desidera accedere ad una risorsa, detta Service Equipment (SE), protetta dal Server AAA del Service Provider (SP) erogatore. Nell'effettuare il controllo di accesso, il SP

Capitolo 2 Framework AAA

può aver bisogno della partecipazione della User Home Organization (UHO), responsabile per l'utente. Il diagramma in figura 2.1 mostra i principali attori in gioco nel modello ed i rispettivi accordi di fiducia e di servizio.

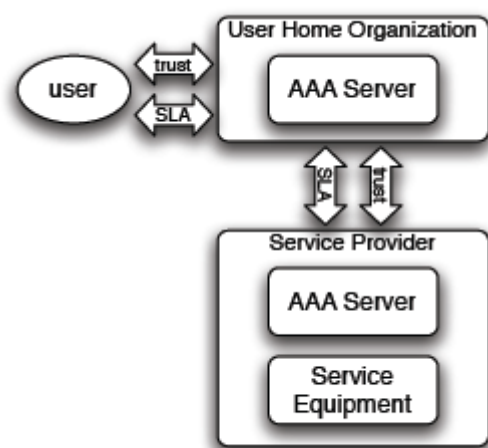


Figura 2.2 Macro-componenti e accordi di servizio

L'autorizzazione dell'utente si fonda su accordi di trust tra il dominio responsabile dell'erogazione del servizio (SP) e quello responsabile per l'utente (UHO). Quest'ultimo può partecipare attivamente, con funzioni di mediatore, alle interazioni di autorizzazione. A seconda del numero di domini coinvolti, si individuano tre casistiche distinte:

4. Single Domain Case: la UHO non viene coinvolta nelle interazioni od è inglobata nel SP.
5. Roaming: la UHO è distinta dal SP e partecipa all'autorizzazione.
6. Distributed Services: è il caso più generale, in cui il servizio richiesto dall'utente è il risultato della composizione di servizi erogati da più SP.

Capitolo 2 Framework AAA

Analizziamo più in dettaglio queste tre casistiche.

2.3.1 Single Domain

Gli attori coinvolti in questo caso sono l'utente, l'AAA Server del SP ed il SE del SP. [4] considera tre possibili sequenze di interazioni, denominate agent, pull e push.

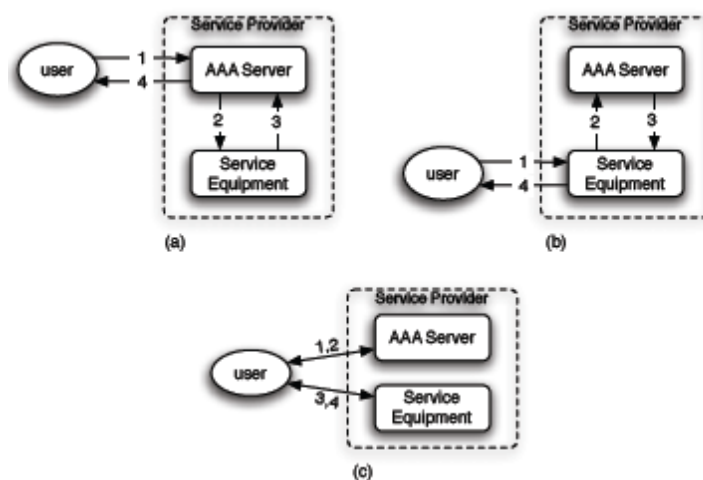


Figura 2.3 Le tre sequenze di interazioni

Agent Sequence. Nella sequenza agent, illustrata in figura 2.2 (a), l'AAA Server funge da intermediario (agente) tra l'utente ed il SE. L'AAA Server riceve la richiesta di servizio dell'utente (1) e, dopo averlo autorizzato, la instrada verso il SE (2). Ricevuta la risposta dal SE (3), l'AAA Server la invidia all'utente (4).

Capitolo 2 Framework AAA

Pull Sequence. Nella sequenza pull (fig. 2.3.(b)) l'utente interagisce direttamente con il SE, come se non vi fosse autorizzazione. È il SE che, una volta ricevuta una richiesta dall'utente (1), provvede ad interrogare il servizio di AAA (2) per ottenerne l'autorizzazione. L'AAA Server effettua la decisione di accesso e ne comunica l'esito al SE (3). In caso di esito positivo, il SE può ora fornire il servizio richiesto all'utente (4).

Push Sequence. La sequenza push (fig. 2.3 (c)) prevede che sia l'utente (per l'esattezza lo user agent) a trasportare l'autorizzazione dal decisore (l'AAA Server) all'erogatore (il SE). Ciò avviene attraverso una prima interazione con l'AAA Server (1) attraverso la quale l'utente ottiene un token¹⁰ (2) che gli garantisce l'accesso al servizio. In un istante successivo, ma non necessariamente correlato alle interazioni (1,2), l'utente può fare uso di tale token (anche più di una volta) per accedere ai servizi del SE (3,4).

2.3.2 Roaming

In molti scenari applicativi avanzati il controllo di accesso è un'operazione complessa che richiede la collaborazione di più domini di competenza. Il caso del roaming prevede la distribuzione delle competenze di autorizzazione tra la UHO ed il SP. La prima, infatti, ha una maggiore conoscenza dell'utente e può quindi basare la sua decisione di accesso su informazioni sul suo conto non disponibili (ad es. per motivi di privacy) al SP. D'altro canto il SP dispone di tutte le informazioni relative ai SE di propria responsabilità e le può utilizzare per effettuare la propria (parte di) autorizzazione.

Le sequenze di interazioni agent, pull e push relative al caso Single Domain sono

¹⁰ Da intendersi nell'accezione generica di documento asserente l'attitudine dell'utente ad usufruire del servizio. Per garantirne la non falsificabilità tale documento dovrà essere firmato(elettronicamente) dall'AAA Server.

Capitolo 2 Framework AAA

generalizzabili al caso Roaming come descritto in figura 2.4. Per semplificare la trattazione di tali sequenze generalizzate, l'AAA Server ed il SE del SP sono stati accorpati. Si noti tuttavia che nella pratica sarà uno dei due componenti del SP (od entrambi) a gestire le singole interazioni con la UHO e l'utente.

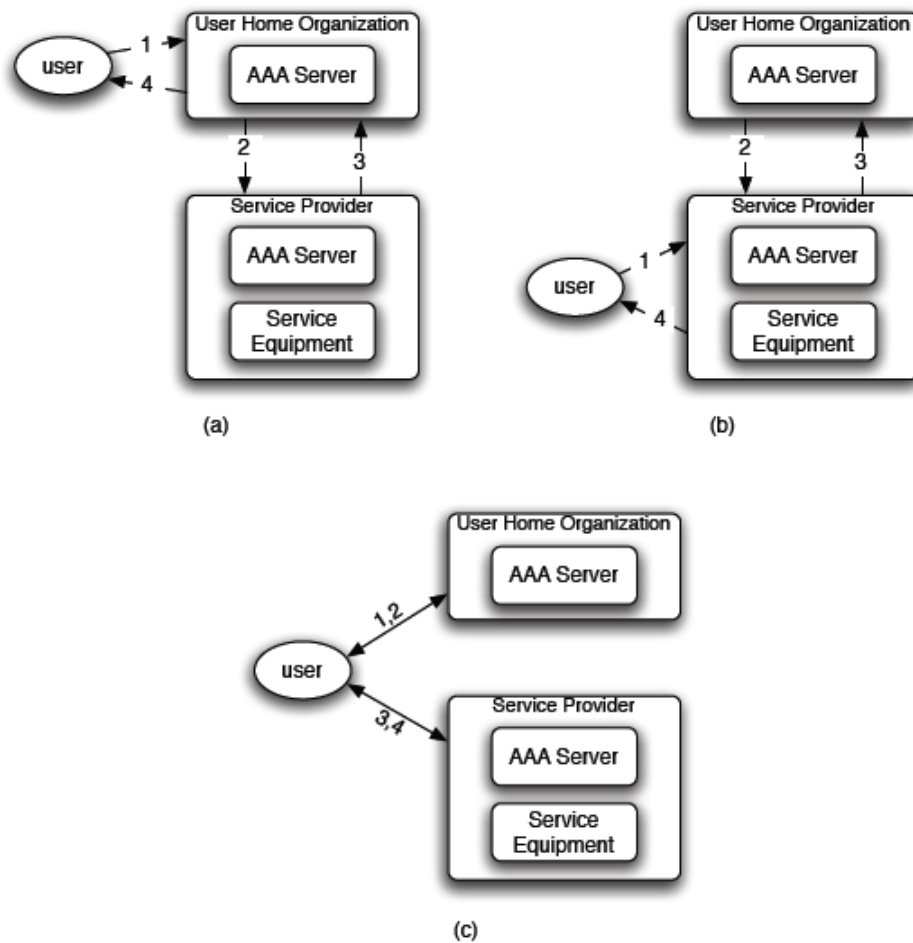


Fig 2.4 Sequenze agent(a), pull(b) e push(c) nel caso del roaming

2.3.3 Distributed Services

In questo caso, il servizio richiesto dall'utente può essere il risultato della cooperazione di più SE locati presso SP distinti, ciascuno con una propria autorità in termini di autorizzazione. Per i Distributed Services si possono combinare gli stessi schemi visti nei casi precedenti, risultanti in molti sotto-casi possibili. Le relazioni 'contrattuali' tra i domini, nel caso di due providers, sono riportate in figura 2.5.



Fig 2.5 Distributed Services

2.4 Policies per l'autorizzazione

Un altro aspetto importante delle problematiche di controllo di accesso è la rappresentazione delle regole che determinano le decisioni di accesso effettuate dagli AAA Servers dei vari providers. L'insieme di tali regole viene spesso denominato policy.

Secondo quanto riportato in [4] l'autorizzazione può essere vista come il risultato della valutazione delle policies di tutte le organizzazioni che hanno interesse nella decisione. Al fine di implementare una tale visione 'policy-centric' è necessario un framework che permetta di:

- recuperare le policies da opportune basi di dati (Policy Retrieval);
- valutare tali policies per ottenere una decisione di autorizzazione (Policy Evaluation);
- applicare la decisione ottenuta alla particolare istanza di accesso (Policy Enforcement);

Capitolo 2 Framework AAA

- specificare come le policies possano essere distribuite tra più domini.

In [4] non si specifica un linguaggio particolare per definire le policies, in quanto è una competenza del Policy Working Group di IETF, si veda in proposito [5].

Policy Retrieval. Questa funzione è a carico dell'organizzazione che necessita della policy ed è svolta da un componente logico denominato Policy Retrieval Point (PRP). Esso si occupa di indicizzare le policies in base al richiedente (l'utente o la UHO), al servizio o ad altri tipi di chiave e di scegliere la policy pertinente alla specifica decisione di accesso.

Policy Evaluation. La funzione di autorizzazione vera e propria è svolta dal Policy Decision Point (PDP) e può essere effettuata in più domini distinti.

La valutazione della policy necessita delle informazioni determinanti l'esito dell'accesso come lo stato della risorsa richiesta, l'identità dell'utente o condizioni ambientali (ora di accesso, locazione, route.).

Il PDP può servirsi di un altro componente, il Policy Information Point (PIP), per ottenere tali informazioni da opportune fonti, in maniera sincrona (pull) od asincrona (popolazione per eventi). In alternativa, l'organizzazione che necessita della decisione può inviare la propria policy e farla valutare direttamente ai PDP delle organizzazioni che detengono le informazioni necessarie.

Capitolo 2 Framework AAA

Policy Enforcement. L'attuazione della decisione emessa dal/dai PDP è tipicamente effettuata dall'organizzazione che protegge i SE, ovvero il SP, o dai SE stessi. Il componente logico che svolge tale funzione viene denominato Policy Enforcement Point (PEP).
Modello di distribuzione dei componenti. I vari componenti logici qui definiti (PRP, PDP, PIP, PEP) possono essere distribuiti in vari punti dell'architettura generale descritta in figura 2.2 nonché replicati in più istanze, eccezion fatta per il PEP. Tali componenti sono gli elementi costitutivi dell'AAA Server, ma possono essere integrati anche all'interno dello user agent e dei SE.

3.5 Autenticazione AAA

Autenticazione di solito significa che c'è qualche modo per assicurare che l'entità con cui stai parlando è chi dice di essere. Questo è chiamato autenticazione di un canale end-point. Di solito si ha bisogno anche di autenticare te stesso al servizio per far sì che il servizio sia sicuro che tu sia tu e non qualcun altro che pretende di essere te. Questa è l'autenticazione del messaggio originario.

L'autenticazione può essere basato su diversi tipi di metodi. Il metodo più usato è quello della password, ma esso non è una buona scelta, perché le password sono tipicamente corte e facili da conoscere. Il metodo più sicuro include l'uso della crittografia di una chiave pubblica, uno schema challenge-response o una crittografia simmetrica.

L'autenticazione in generale ha diversi requisiti di sicurezza. Questi includono la protezione contro attacchi di tipo replay o man-in-the-middle e confidenzialità.

3.6 Accounting AAA

Nel processo di accounting vengono raccolti i dati di consumo delle risorse a seconda dei diversi scopi. Questi dati possono essere usati, per esempio, per un'analisi di tendenza, per il conto, o per l'auditing. Ognuno di essi ha diversi requisiti. I requisiti dipendono anche dal fatto se sono intra domain accounting o inter domain accounting. Nel primo caso le informazioni di accounting non attraversano i confini amministrativi. Nel secondo caso le informazioni di accounting attraversano i confini amministrativi ed sono, perciò, più suscettibili di violazioni della sicurezza[6].

Un inter domain accounting ha bisogno di occuparsi della perdita di pacchetti e dei requisiti di sicurezza alternativi. Così ci dovrebbe essere bisogno di una risposta di protezione, non ripudiabilità, integrità dei dati e confidenzialità oltre all'autenticazione.

3.6.1 Analisi dell'andamento e pianificazione di capacità

Nell'analisi dell'andamento e nella pianificazione di capacità la perdita di alcuni pacchetti non è un problema, perché essi sono usati per valutare l'uso futuro delle risorse. Quindi è richiesta una robustezza moderata contro la perdita di pacchetti nel caso di un intra domain e la robustezza contro un'alta perdita di pacchetti nel caso dell'inter domain. Questo perché l'affidabilità del trasferimento dati nel caso di un intra domain e di un inter domain sono differenti. Inoltre l'integrità, l'autenticazione e la risposta di protezione sono richiesti così come la confidenzialità nel caso dell'inter domain.

3.6.2 Conto

Il conto può essere sia a uso che non a uso. Nel caso sia non a uso nessuna informazione di accounting viene richiesta per il conto. In questo caso non c'è danno, anche se tutti i dati di sessione fossero persi. Nel caso sia a uso, comunque, è necessario un approccio archivistico dei dati di accounting. In questo caso il conto ha alcuni vincoli di ritardo, perché è desiderabile minimizzare il rischio finanziario. Come sempre, quando i soldi sono interessati c'è un bisogno di autenticazione, integrità e anche confidenzialità e non ripudio.

3.6.3 Auditing

L'auditing, che è l'atto della verifica della precisione di un processo, ha bisogno dei dati di accounting nella verifica del processo. Questi dati devono essere affidabili e sicuri quanto i dati che l'entità verificata sta usando.

3.7 Scalabilità e affidabilità dell'accounting

Quando consideriamo l'accounting ci sono tre aree che dovrebbero essere prese in considerazione: tolleranza ai guasti, consumo delle risorse e il modello della raccolta dati.

La tolleranza ai guasti è dovuta al fatto che c'è la preoccupazione dei soldi nel caso dell'accounting. Se il sistema non funziona per alcune ragioni provocherà una perdita di guadagno. Le situazioni tipiche di guasti sono perdita dei pacchetti, guasti alla rete, guasti al server di accounting e il reboot dei dispositivi. Questo può essere gestito in diversi modi. Per esempio può essere usata una memoria non volatile. Maggiori informazioni si possono trovare in [6].

Capitolo 2 Framework AAA

L'accounting ha bisogno di risorse per funzionare completamente. Le più importanti sono la larghezza di banda della rete, la memoria, e la CPU. Ognuno di queste risorse hanno alcuni impatti sull'intero funzionamento del sistema ed essi dovrebbero essere considerati quando si progetta l'accounting. L'uso delle risorse può essere ottimizzato in molti casi usando tecniche appropriate. Per esempio l'uso della larghezza di banda può essere utilizzato usando il batching[6].

Capitolo 3 RADIUS

3.1 Introduzione

RADIUS, letteralmente Remote Authentication Dial In User[1], è un protocollo ampiamente usato e implementato per gestire gli accessi ai servizi della rete. Esso è un'implementazione del framework AAA visto nel capitolo precedente, anche se quest'ultimo è stato implementato sulla base del protocollo RADIUS. Il protocollo RADIUS definisce uno standard per lo scambio di informazioni tra il NAS¹¹ e un server AAA per consentire le operazioni di autenticazione, autorizzazione e accounting. Un server RADIUS può gestire profili utenti per l'autenticazione, verificando il nome utente e la password, informazioni di configurazione, che specificano il tipo di servizio da inviare, e le policy da attuare per poter restringere l'accesso all'utente.

RADIUS è un'architettura client/server per l'autenticazione, l'autorizzazione e l'accounting. Il client RADIUS è costituito da un server di accesso, il NAS, che può essere ad esempio un dial-up server, un firewall o un access point. Il client invia le credenziali dell'utente e i parametri di connessione, oltre a tutti i messaggi contenenti informazioni di accounting, sotto forma di messaggi RADIUS al server. Il server Radius autentica e autorizza le richieste provenienti dal client ed è interfacciato ad una directory o ad un database dove sono definiti gli utenti e le politiche di accesso.

Nella figura 3.1 viene riportato il principio di funzionamento del sistema di autenticazione.

¹¹ verrà definito anche client

Capitolo 3 RADIUS



Figura 3.1 Principio di funzionamento sistema di autenticazione

I server RADIUS sono responsabili della ricezione delle richieste di connessione dell'utente, dell'autenticazione di quest'ultimo, e inviano, poi, tutte le informazioni di configurazioni necessarie al client per inviare i servizi all'utente.

Il server RADIUS può essere configurato per autenticare una richiesta localmente o agire come un client proxy e inviare una richiesta ad un altro server.

3.2 Stabilire una sessione RADIUS

La richiesta di un utente per stabilire una connessione alla rete è una insieme di scambi di messaggi. Descriviamo brevemente in questo paragrafo come avviene la richiesta e l'eventuale concessione dell'accesso.

Quando un utente si connette al client, il client invia un pacchetto chiamato Access-Request al server. Quando il server riceve la richiesta, esso valida il client inviante. Se al client è permesso inviare richieste al server, il server prenderà

Capitolo 3 RADIUS

allora le informazioni dall'Access-Request e cercherà di associare la richiesta al profilo utente. Il profilo conterrà una lista di requisiti che devono essere soddisfatti per autenticare l'utente. Nel processo di autenticazione viene coinvolta, di solito, la verifica della password, ma possono essere verificate, anche, altre informazioni, come il numero della porta del client o il tipo di servizio che è stato richiesto.

Se tutte le condizioni sono soddisfatte, il server invierà un pacchetto Access-Accept al client; altrimenti verrà inviato un pacchetto Access-Reject. I dati di un pacchetto Access-Accept includono le informazioni di autorizzazione che specificano per quali servizi l'utente può avere accesso alla rete e altre informazioni di sessione, come il valore di timeout che indicherà quando l'utente dovrà essere disconnesso dal sistema.

Quando il client riceve un pacchetto Access-Accept, esso genererà un Accounting-Request per iniziare la sessione e invierà la richiesta al server. I dati del pacchetto Accounting-Request descrivono il tipo di servizio richiesto e l'utente che utilizzerà questo servizio. Il server risponderà con un Accounting-Response per confermare che la richiesta è stata ricevuta con successo e registrata. La sessione dell'utente terminerà quando il client genererà un Accounting-Request per terminare la sessione. Il server, infine, confermerà l'Accounting-Request con un Accounting-Response.

Capitolo 3 RADIUS

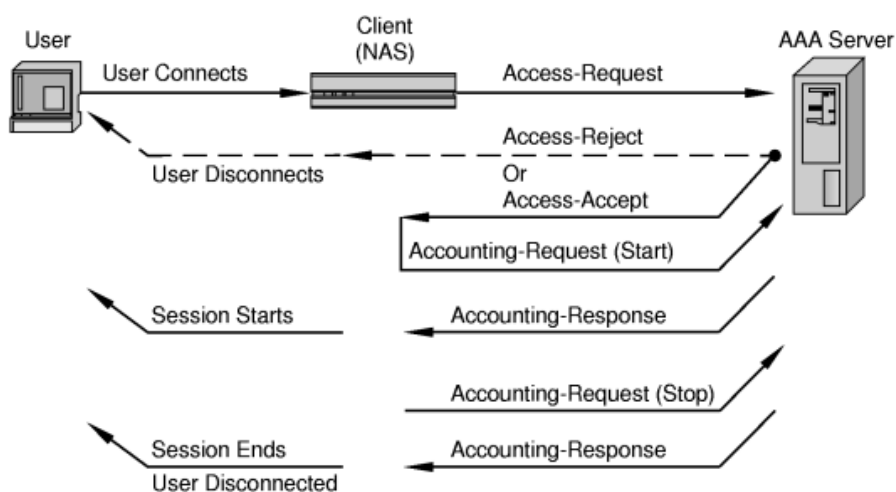


Fig 3.2 Sequenza di connessione

3.3 Caratteristiche del protocollo RADIUS

Il protocollo RADIUS usa come protocollo di trasporto l'UDP. Il motivo della scelta dell'uso di questo protocollo non affidabile rispetto all'utilizzo del più affidabile protocollo TCP è semplicemente una scelta tecnica.

Spieghiamo alcuni concetti da capire per comprendere questa scelta.

RADIUS è un protocollo di transazione che ha diverse caratteristiche:

1. se la richiesta al server di autenticazione fallisce, deve essere richiesto un secondo server. Per soddisfare questo requisito, deve essere mantenuta una copia della richiesta sul livello trasporto per permettere una trasmissione alternativa. Questo richiede la presenza di un timer di ritrasmissione.
2. I requisiti di sincronizzazione di questo particolare protocollo sono significativamente diversi da quelli forniti dal protocollo TCP. Ad un

Capitolo 3 RADIUS

estremo, il RADIUS non richiede un reattivo rilevamento dei dati persi. L'utente è disposto ad aspettare diversi secondi per completare l'autenticazione. La generale aggressività della ritrasmissione TCP (basato sul tempo medio di round trip) non è richiesta, né lo è l'overhead dovuto all'invio di ack del TCP. Dall'altra parte, l'utente non è disposto ad aspettare diversi minuti per l'autenticazione. Perciò l'invio affidabile dei dati TCP due minuti più tardi non sarebbe utile. Il più veloce uso di un server alternativo permette all'utente di guadagnare l'accesso prima di rinunciare.

3. La natura stateless di questo protocollo semplifica l'uso dell'UDP. Il client e il server vanno e vengono. Con l'UDP è possibile per un client e un server aprire le loro sessioni solo una volta e lasciarle aperte anche quando si verificano guasti sulla rete.
4. UDP semplifica le implementazioni server. Nelle prime implementazioni il server era single thread. Questo significa che una singola richiesta veniva ricevuta, processata e rispedita. Questo tipo di implementazione non era gestibile in ambienti dove il meccanismo di sicurezza back-end deve essere in tempo -reale (si parla di secondi). La coda di richiesta del server si riempirebbe e in ambienti dove, ogni minuto, centinaia di persone devono essere autenticate, esso richiede che il tempo di risposta sia maggiore di quanto gli utenti sono disposti ad attendere. La soluzione ovvia era quella di rendere il server multi-threaded. Per realizzare questo era indispensabile l'UDP. Sono stati, così generati processi separati per servire ogni richiesta e questi processi dovrebbero rispondere direttamente al client con un semplice pacchetto UDP.

Anche con il protocollo UDP i problemi non sono del tutto risolti. L'uso di UDP richiede una cosa che è costruita nel TCP: con UDP dobbiamo artificialmente

Capitolo 3 RADIUS

gestire i timer per la ritrasmissione allo stesso server, anche se non richiedono la stessa attenzione alla sincronizzazione fornita dal TCP.

3.4 Metodi di autenticazione

Radius supporta diversi protocolli di autenticazione per inviare informazioni sensibili dell'utente da e per il server di autenticazione. I principali metodi sono il PAP (Password Authentication Protocol), il CHAP (Challenge Handshake Authentication Protocol) e l'EAP (Extensible Authentication Protocol), utilizzato all'interno di 802.1x.

3.4.1 PAP e CHAP

PAP

Il protocollo di autenticazione PAP è stato definito inizialmente per essere usato con il protocollo PPP¹² in [7].

PAP fornisce un semplice metodo di autenticazione tra le peer per stabilire la propria identità usando il metodo 2-way handshake. Questo viene fatto solo all'inizio della fase Link Establishment, dove viene scelto il metodo di autenticazione.

Una volta completata questa fase, dalla peer¹³ viene inviata una coppia username-password all'authenticator¹⁴ finché l'autenticazione non viene confermata o la connessione viene terminata.

PAP non è un metodo di autenticazione forte. Le password vengono inviate in

¹² Point-to-Point protocol RFC 1661

¹³ Entità finale di un link PPP: entità che viene autenticata

¹⁴ Entità finale di un link PPP che richiede l'autenticazione

Capitolo 3 RADIUS

chiaro e non ci sono protezioni, per esempio, contro attacchi sniffing.

Quando viene usato con RADIUS per l'autenticazione, i messaggi scambiati tra il client e il server per stabilire una connessione PPP corrisponde alla figura 3.2. Questo metodo di autenticazione è molto usato quando una password in chiaro deve essere disponibile per simulare un login ad un host remoto. In questo caso, questo metodo fornisce un livello di sicurezza simile alla login di utente all'host remoto.

L'attributo User-Password in un pacchetto Access-Request segnala al server RADIUS che il protocollo PAP sarà usato per quella transazione. È importante notare che il solo campo richiesto in questo caso è il campo User-Password. Il campo User-Name non deve essere incluso nel pacchetto Access-Request ed è probabile che il server RADIUS durante la catene di proxy cambierà il valore nel campo User-Name.

L'algoritmo usato per nascondere la password originale dell'utente è composto da molti elementi. Primo, il client individua l'identificativo e la shared secret per la richiesta originale e sottopone esso ad una sequenza di hashing MD5. La password originale del client viene sottoposta ad un processo di Xor e il risultato proveniente da queste due sequenze è poi messo nel campo User-Password. Il server RADIUS ricevente fa l'operazione inversa di queste procedure per determinare se autorizzare la connessione. La natura del meccanismo di nascondere le password previene l'utente da determinare se, quando fallisce l'autenticazione, l'insuccesso è stato causato da una password non corretta o da un secret non valido.

PAP non è un metodo EAP ed è implementato sono nel tunnelled protocol EAP-TTLS in cui il tunnel protegge le password in chiaro.

Capitolo 3 RADIUS

CHAP

Come il protocollo PAP, anche il protocollo CHAP [8] è stato definito per essere usato con il protocollo PPP.

Il protocollo CHAP è usato per verificare periodicamente l'identità della peer usando un 3-way handshake. Questo viene fatto dopo la fase di Link Establishment.

Dopo aver completato questa prima fase, l'authenticator invia un messaggio challenge alla peer. Essa risponde con un valore calcolando una funzione di hash one-way. L'authenticator controlla la risposta con il valore di hash calcolato da lui. Se i valori coincidono, l'autenticazione viene riconosciuta; altrimenti la connessione viene terminata. Dopo un intervallo di tempo casuale, l'authenticator invia un nuovo messaggio challenge e si ricomincia con il controllo.

Ci sono molti vantaggi nell'uso del protocollo CHAP. CHAP fornisce protezione contro attacchi di playback attraverso l'uso di un identificativo incrementale e un valore challenge variabile. Il fatto che viene richiesta una ripetizione della challenge è destinato a limitare il tempo di esposizione ad ogni singolo attacco.

Questo metodo dipende da una secret conosciuta solo dall'authenticator e dalla peer. La secret non viene inviata lungo il link.

Anche se l'autenticazione è unidirezionale, negoziando la stessa secret in entrambe le direzioni può essere facilmente usato per l'autenticazione reciproca.

Dato che CHAP può essere usata per autenticare molti sistemi, il campo name può essere usato come indice per localizzare la propria secret in una grande tabella di secret. Questo rende possibile il supporto di più di una coppia name/secret per sistema, e permette di cambiare la secret in uso in ogni momento della sessione.

CHAP ha anche alcuni svantaggi. CHAP richiede che la secret sia disponibile in chiaro. Il protocollo CHAP non può essere utilizzato per grandi installazioni, poiché ogni secret viene conservata su entrambe le entità finali del link.

Quando è usato con RADIUS per l'autenticazione, i messaggi scambiati tra il

Capitolo 3 RADIUS

client e il server per stabilire una connessione PPP è simile alla figure 3.2. una differenza, comunque, è che la challenge avviene tra l'utente e il NAS prima che il NAS invii un Access-Request. L'utente deve rispondere cifrando la challenge e inviando il risultato. Gli utenti autorizzati sono muniti di dispositivi speciali, come smart card, che possono calcolare la risposta corretta. Il NAS invierà, allora, la challenge e la risposta nell'Access-Request, che il server AAA userà per autenticare l'utente.

Anche il protocollo CHAP, come PAP, non è un metodo EAP ed è implementato solo nel tunnelled protocol EAP-TTLS.

In questo protocollo la password non dovrebbe mai essere inviata in alcun pacchetto lungo la rete. CHAP cifra dinamicamente l'ID della richiesta dell'utente e la password. La macchina dell'utente, allora, completa la sua procedura di logon, ottenendo una chiave di dispositivi RADIUS client di almeno 16 ottetti. Il client, allora crea l'hash di questa chiave e invia al server un CHAP ID, un CHAP response e lo username del client RADIUS. Quest'ultimo, una volta ricevuto il tutto, pone il campo CHAP-ID nell'appropriato posto nell'attributo CHAP-Password e poi invia la risposta. Il valore di challenge originariamente ottenuto è messo o nell'attributo CHAP-Challenge o nel campo Authenticator nell'header; questo fa in modo che il server possa accedere facilmente al valore per autenticare l'utente.

Per autenticare l'utente, il server RADIUS usa il valore CHAP-Challenge, il CHAP-ID e la password sul record per quel particolare utente e lo sottopone ad un altro meccanismo di hash. Il risultato di questo algoritmo dovrebbe essere identico al valore trovato nell'attributo CHAP-password. Se non è così, il server deve negare la richiesta; altrimenti la richiesta è concessa.

Il fatto che la password, nelle transazioni CHAP non attraversa mai la rete è una ragione del perché CHAP è un protocollo di autenticazione interessante.

I dati dell'utente a cui viene applicata la procedura di hash restituisce un valore

Capitolo 3 RADIUS

one-way che non contiene la password. Così il server deve avere la password dell'utente attuale memorizzata in chiaro sul proprio record per creare un hash con cui confrontarla. Gli ID CHAP non sono persistenti; questo fa sì che si riduca la possibilità che terze parti sniffino o che si introducano nella trasmissione. In più, il protocollo CHAP supporta il challenging il client in qualsiasi momento durante la sessione dell'utente, il che aumenta la possibilità che utenti non validi siano tenuti fuori dal sistema.

3.4.4 802.1x

L'IEEE 802.1X[9] offre un framework per l'autenticazione e il controllo del traffico degli utenti per una protezione della rete, oltre alle chiavi di crittografia che variano dinamicamente. 802.1x lega il protocollo EAP[10] ai dispositivi delle reti LAN sia wireless che cablate e supporta diversi metodi di autenticazione, come token card, Kerberos, one-time password, certificati e l'autenticazione tramite chiave pubblica.

Nell'architettura 802.1x, ci sono tre componenti chiavi:

1. il supplicant: l'utente o il client che vuole essere autenticato;
2. il server di autenticazione di solito un server RADIUS;
3. l'authenticator: il dispositivo in mezzo, come un access point.

Capitolo 3 RADIUS

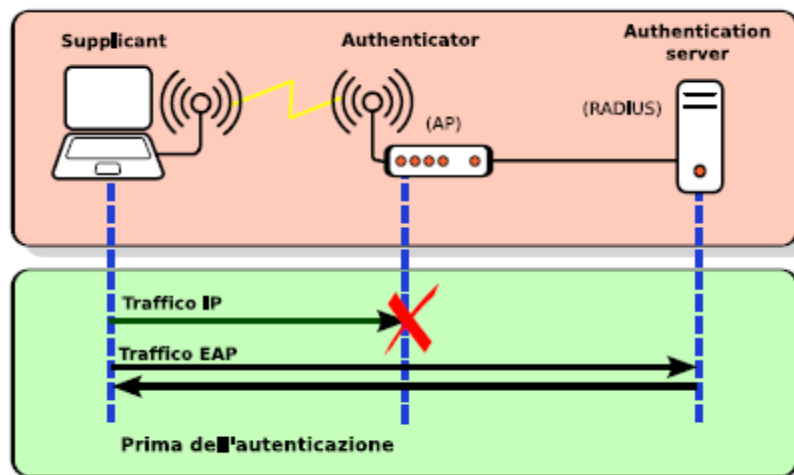


Fig. I tre componenti dell'architettura 802.1x

Il protocollo chiave in 802.1x è chiamato EAP over LANs (EAPoL). Attualmente è definito per LAN come Ethernet incluse le reti wireless 802.11.

Il processo di funzionamento dell'802.1x è il seguente:

- il supplicant invia un pacchetto EAP-Response/Identity all'authenticator (per esempio un access point 802.11), che viene poi passato al server di autenticazione (il server RADIUS che è situato nella parte dell'access point cablato).
- Il server di autenticazione invia un challenge all'authenticator. Quest'ultimo sfascia la challenge dall'IP e lo inserisce nell'EAPoL e lo invia al supplicant.
- Il supplicant risponde alla challenge attraverso l'authenticator e passa la risposta al server di autenticazione. Esso usa uno specifico algoritmo di autenticazione per verificare l'identità del client. Questo può essere fatto attraverso l'uso di certificati digitali o altri tipi di autenticazione EAP.

Capitolo 3 RADIUS

- Se il supplicant fornisce la corretta identità, il server di autenticazione risponde con un messaggio di successo, che viene inviato al supplicant. L'authenticator apre le porte al supplicant per l'accesso alla LAN sulla base di attributi che provengono dal server di autenticazione.

Il protocollo 802.1x (EAPoL) fornisce l'autenticazione indipendentemente da l'implementazione delle chiavi WEP o senza la crittografia. Se configurato per implementare lo scambio dinamico delle chiavi, il server di autenticazione può inviare una chiave di sessione all'access point con il messaggio di accept. L'access point usa la chiave di sessione per costruire, firmare e cifrare un messaggio chiave EAP che viene inviato al client subito dopo l'invio del messaggio di successo. Il client può usare il contenuto del messaggio chiave per definire le chiavi di crittografia applicabili.

802.1x è un meccanismo di invio e non fornisce meccanismi di autenticazione. Nell'utilizzare l'802.1x, si ha bisogno di scegliere un tipo di EAP, come l'EAP-TLS (EAP-Transport Layer Security) o EAP-TTLS (EAP-Tunnelled TLS), che definisce come avviene l'autenticazione. Questi sono quelli che, tramite l'uso di tunnel cifrati con SSL/TLS, danno maggiori garanzie di sicurezza e supporto per la generazione e lo scambio di chiavi crittografiche utilizzate in WPA e 802.11i.

Il tipo EAP risiede sul server di autenticazione e all'interno del sistema operativo o applicazioni software sui dispositivi client. L'access point agisce come un pass-through per i messaggi 802.1x, il che significa che è possibile specificare alcuni tipi di EAP senza avere la necessità di upgrade l'access point.

L'EAP-TLS necessita di un certificato digitale X.509 con relativa chiave privata sia sul RADIUS server che sul supplicant. Spesso, data la difficoltà con cui si riesce a dotare un'organizzazione di una PKI X.509 in grado di fornire ad ogni utente un

Capitolo 3 RADIUS

certificato, si rinuncia a utilizzare EAP-TLS, anche se questo metodo è sicuramente quello più sicuro poiché l'utente non deve digitare username e password.

Del resto EAP-TLS è ideale quando si vuole autenticare l'accesso in rete mediante smart card memorizzando la chiave privata e il certificato dell'utente direttamente su tale supporto.

Il *PEAP* (Protected EAP) così come l'*EAP-TTLS* necessitano di un certificato X509 e della relativa chiave privata solo dal lato del RADIUS server. Con tale certificato viene autenticato l'authentication server nei confronti del client e successivamente stabilito un tunnel SSL/TLS in cui far passare un altro protocollo con cui autenticare l'utente nei confronti di RADIUS. In genere tale protocollo è Ms-CHAPv2 in cui l'utente deve inserire username e password. Tra il PEAP e l'EAP-TTLS il PEAP con MS-CHAPv2 è quello più utilizzato. Ciò, probabilmente, è dovuto al fatto che Windows XP dà supporto nativo per questo protocollo. Se invece si vuole utilizzare EAP-TTLS su Windows bisogna ricorrere ad un supplicant di terzi. Per quanto riguarda Linux, invece, Xsupplicant supporta tutti e tre i metodi di autenticazione.

Capitolo 4 RADIUS: Pacchetti e Attributi

In questo capitolo verranno approfondite le caratteristiche dei diversi pacchetti che sono utilizzati dal protocollo RADIUS e le diverse tipologie di attributi che sono riportati dai pacchetti per aggiungere maggiori potenzialità al protocollo RADIUS.

4.1 Formato dei pacchetti

I dati tra il client e il server vengono scambiati all'interno dei pacchetti RADIUS. Questi pacchetti, come detto nel paragrafo precedente, vengono incapsulati nel protocollo di trasporto UDP. Il protocollo comunica sulla porta di destinazione 1812.

Il formato del pacchetto RADIUS è mostrato in figura 4.1

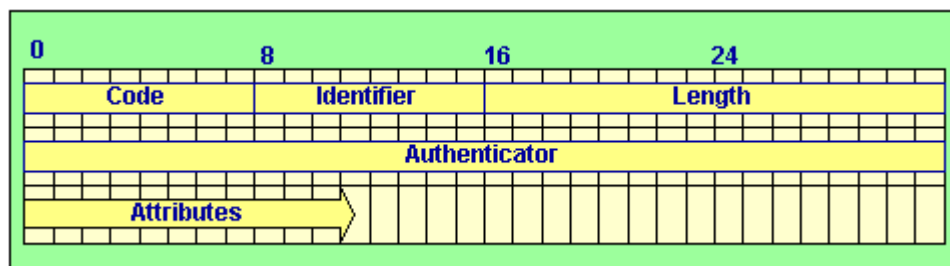


fig 4.1 formato del pacchetto RADIUS

Campo Code

Capitolo 4 RADIUS: Pacchetti e Attributi

Il campo Code, come si può vedere dalla figura 4.1, è un ottetto e identifica il tipo di pacchetto inviato. Quando un pacchetto viene ricevuto con un campo Code non valido, esso viene silenziosamente scartato¹⁵

I codici di questo campo sono i seguenti:

1	Access-Request
2	Access-Accept
3	Access-Reject
4	Accounting-Request
5	Accounting-Response
11	Access-Challenge
12	Status-Server (experimental)
13	Status-Client (experimental)
255	Reserver

Parleremo più in dettaglio di questi tipi di pacchetti nel prossimo paragrafo.

Campo Identifier

Anche questo campo è un ottetto e aiuta nel far corrispondere le richieste con le risposte. Il server può rilevare una richiesta duplicata se in un piccolo periodo di tempo si presentano più richieste con lo stesso indirizzo IP sorgente, stesso numero di porta e l' Identifier.

¹⁵ 1 con l'uso di questa frase si intenderà che il pacchetto viene scartato senza ulteriori processi, e senza informare il client

Capitolo 4 RADIUS: Pacchetti e Attributi

Campo Length

il campo Length, invece, è di due ottetti. Esso indica la lunghezza totale del pacchetto, la quale include anche tutti gli altri campi del pacchetto.

Le specifiche RFC richiedono alcune azioni del server RADIUS in merito alla lunghezza non corretta dei dati. Se il server Radius riceve una trasmissione con un messaggio più lungo del range del campo Length, deve essere trattato come riempitivo e viene ignorato al momento della ricezione. Contrariamente, se il server riceve un messaggio più corto della lunghezza riportata del campo length, il messaggio sarà scartato senza notificazioni.

Campo Authenticator

Questo campo è di sedici ottetti. Il campo *authenticator*, è di sedici ottetti. Questo valore è usato per autenticare le risposte provenienti dal server RADIUS, ed è usato anche nell'algoritmo per nascondere le password.

Ci sono due tipi specifici di valori di autenticazione: Request-Authenticator e Response-Authenticator.

- Request-Authenticator: nei pacchetti Access-Request, il valore Authenticator è un numero random di sedici ottetti. Il valore dovrebbe essere casuale e unico durante il tempo di vita della secret (la password condivisa tra il client e il server RADIUS), dato che la ripetizione di un valore di richiesta insieme alla stessa secret permetterebbero ad un attacker di rispondere con una risposta precedentemente intercettata. Il valore Request-Authenticator in un pacchetto Access-Request dovrebbe essere imprevedibile, infatti un attacker può ingannare un server rispondendo ad una prevista richiesta futura, e usare poi la risposta per passare dal server in una futura Access-Request. Anche se protocolli come RADIUS sono

Capitolo 4 RADIUS: Pacchetti e Attributi

incapaci di proteggersi contro il furto di una sessione di autenticazione che avviene tramite attacchi di intercettazione attiva in real-time, la generazione di una richiesta unica e imprevedibile può proteggere contro un ampio range di attacchi attivi di autenticazione. Il NAS e il server RADIUS condividono una secret. Questa shared secret seguita dal Request Authenticator viene sottoposto ad una funzione di hash MD5 per creare un valore digest di sedici ottetti che a sua volta viene sottoposto alla funzione di Xor con la password inserita dall'utente, e il risultato viene messo nell'attributo User-Password del pacchetto Access-Accept.

- *Response authenticator*: questo campo è il valore del campo Authenticator nei pacchetti Access-Accept, Access-Reject e Access-Challenge. Il valore è calcolato usando la funzione di hash MD5 tra i valori dei campi Code, Identifier, Length e Request-Authenticator dell'header del pacchetto, seguito dal payload del pacchetto e dalla chiave condivisa.

$ResponseAuth = MD5(Code + ID + Length + RequestAuth + Attributes + Secret)$

Campo Attributes

Campo di lunghezza variabile, contiene una lista di Attributi che sono richiesti per il tipo di servizio. Questo campo verrà analizzato più approfonditamente nei paragrafi successivi.

4.2 Tipi di pacchetti

La comunicazione RADIUS usa il paradigma della Request-Response, la richiesta viene fatta dal client e inviata al server, la risposta viene fatta dal server e inviata al client. Possibili coppie Request-Response sono:

Capitolo 4 RADIUS: Pacchetti e Attributi

1. Access-Request (client ->server) la richiesta di accesso per un utente verso alcuni servizi. Le possibili risposte sono:
 - Access-Accept(server->client) risposta positiva ad un Access-Request del client
 - Access-Reject(server->client)risposta negativa a un Access-Request del client
 - Access-Challenge(server->client)risposta su un Access-Request, dove il server si aspetta una risposta dal client incapsulata in un Access-Request.
2. Accounting Request(client->server) richiesta di memorizzare i dati di accounting all'interno del pacchetto sul server. Il risposta è:
3. Accounting Response (server->client) risposta al client quando i dati di accounting sono memorizzati con successo sul server.

I tipi di pacchetti RADIUS sono determinati dal campo Code nel primo ottetto del pacchetto RADIUS.

I tipi di pacchetti RADIUS importanti per le fasi di autenticazione e autorizzazione della transazione AAA sono quattro:

- Access-Request
- Access-Accept
- Access-Reject
- Access-Challenge

Per quanto riguarda l'accounting i tipi di pacchetti importanti sono due:

Capitolo 4 RADIUS: Pacchetti e Attributi

- Accounting-Request
- Accounting-Response

Il pacchetto *Access-Request* è inviato ad un server RADIUS, e comunica le informazioni usate per determinare se un utente ha il permesso di accedere ad uno specifico NAS e tutti i servizi speciali richiesti per quell'utente. In altre parole l'Access-Request è inviata dall'utente quando vuole richiedere un particolare servizio della rete.

Un implementazione per autenticare un utente deve trasmettere un pacchetto RADIUS con il campo Code settato a 1 (il codice decimale che indica l'Access-Request).

Su ricezione di un Access-Request da un client valido, deve essere trasmessa un'appropriata risposta, sia che essa sia di autorizzazione o di rifiuto.

Un Access-Request dovrebbe contenere un attributo User-Name, per identificare la persona che vuole guadagnare l'accesso alle risorse della rete . Esso deve contenere o l'attributo NAS-IP-Address o l'attributo NAS-Identifier o entrambi.

Una Access-Request deve contenere o l'attributo User-Password o quello CHAP-Password o l'attributo State. Un Access-Request non deve contenere entrambi gli attributi riguardanti la password.

Un'Access-Request dovrebbe contenere l'attributo NAS-Port o l'attributo NAS-Port-Type o entrambi a meno che l'accesso richiesto non coinvolga una porta o il NAS non riesce a distinguere le sue porte. Un Access-Request può contenere degli attributi addizionali come suggerimenti, che il server non sempre dovrà ascoltare.

Quando è presente una User-Password, essa viene nascosta utilizzando la funzione di hash MD5.

Capitolo 4 RADIUS: Pacchetti e Attributi

I pacchetti *Access-Accept* sono inviati dal server RADIUS al client per confermare che la richiesta è stata accettata. Questo pacchetto fornisce le informazioni di configurazione necessarie affinché i servizi vengano inviate agli utenti. Se tutti i valori del campo *Attributes* ricevuti in un *Access-Request* sono accettabili allora il server RADIUS deve trasmettere un pacchetto con il campo *Code* settato a 2 (cioè *Access-Accept*). Sulla ricezione di un *Access-Accept*, il client confronta il campo *Identifier* del pacchetto *Access-Accept* con quello del pacchetto *Access-Request* che è stata sospesa in attesa di una risposta. I due campi devono corrispondere per far sì che la risposta riguardi proprio quella richiesta. Il campo *Response Authenticator* deve contenere la risposta corretta per l'*Access-Request* in sospeso. I pacchetti non validi vengono silenziosamente scartati.

Il server RADIUS invia il pacchetto *Access-Reject* al client se deve negare i servizi richiesti. In questo caso nessun valore inserito nel campo *Attributes* viene accettato, il server RADIUS deve trasmettere un pacchetto con il campo *Code* settato a 3 (*Access-Reject*). Il rifiuto può essere basato sulle policy del sistema, su privilegi insufficienti o su altri criteri. L'*Access-Reject* può essere inviato in qualunque momento durante la sessione, questo lo rende perfetto per far rispettare i limiti di tempo di collegamento. Comunque, non tutti dispositivi supportano la ricezione dell'*Access-Reject* durante una connessione prestabilita.

Esso può includere un o più *Attributes Reply-Message* con un messaggio di testo che il NAS può visualizzare all'utente.

Se il server riceve dall'utente informazioni discordanti esso può richiedere più informazioni o se vuole, semplicemente, far diminuire il rischio di autenticazioni fraudolente, esso può emettere un pacchetto *Access-Challenge* al client. Il client, una volta ricevuto l'*Access-Challenge* deve emettere una nuova *Access-Request* con, incluse, le appropriate informazioni. Il campo *Code* di questo tipo di pacchetto verrà settato a 11 (*Access-Challenge*).

Capitolo 4 RADIUS: Pacchetti e Attributi

Il campo Attributes può avere uno o più attributi Reply-Message , e può avere un singolo State Attribute, o nessun attributo.

Al momento della ricezione dell'Access-Challenge il campo Identifier viene confrontato lo stesso campo dell'Access-Request in sospeso. In più il campo Response Authenticator deve contenere la corretta risposta per l'Access-Request in sospeso. I pacchetti non validi sono silenziosamente scartati.

Nel caso in cui il client non supporta il processo di challenge/response esso tratta un Access-Challenge come se avesse ricevuto un Access-Reject. Se il client supporta il processo challenge/response, la ricezione di una valida Access-Challenge indica che una nuova Access-Request dovrebbe essere inviata. Il client può visualizzare il messaggio di testo all'utente e sollecita l'utente ad una risposta. Il client invia la sua Access-Request originale con un nuovo ID di richiesta e il Request Authenticator, con l'attributo User-Password sostituito dalla risposta dell'utente (cifrata) e includendo l'attributo State dell'Access-Challenge. Solo le istanze 0 e 1 dell'attributo State possono essere presenti nell'Access-Request.

Fino a qui abbiamo analizzato i pacchetti che vengono utilizzati nel processo di autenticazione e in quello di autorizzazione. Passiamo ora ad analizzare i due tipi di pacchetti utilizzati per l'accounting.

I pacchetti Accounting-Request sono inviati dal client al server RADIUS e trasmettono le informazioni usate per fornire informazioni di accounting per un servizio fornito all'utente. Il client trasmette un pacchetto RADIUS con il campo code settato a 4(Accounting-Request).

Alla ricezione di un Accounting-Request, il server deve trasmettere una risposta Accounting-Response se registra con successo il pacchetto di accounting.

Capitolo 4 RADIUS: Pacchetti e Attributi

altrimenti, se il server RADIUS fallisce la registrazione del pacchetto di accounting non dovrà trasmettere nessuna risposta al client e il pacchetto verrà silenziosamente scartato..

Ogni attributo valido in un pacchetto RADIUS Access-Request o Access-Accept è valido in un pacchetto RADIUS Accounting-Request, ad eccezione di alcuni attributi, che non devono essere presenti in quest'ultimo pacchetto, come User-Password, Reply-Message o State.

Se il pacchetto Accounting-Request include un Framed-IP-Address, questo attributo deve contenere l'indirizzo IP dell'utente. Se l'Access-Accept usa un valore speciale per il Framed-IP-Address dicendo al NAS di assegnare o negoziare un indirizzo IP per l'utente, il Framed-IP-Address nell'Accounting-Request deve contenere l'attuale indirizzo IP assegnato o quello negoziato.

Il campo Identifier di questo pacchetto deve essere cambiato ogni volta che il contenuto del campo Attributes cambia, e ogni volta una risposta valida viene ricevuta per una richiesta precedente. Per ritrasmissioni dove i contenuti sono identici, l'Identifier non deve cambiare.

È possibile includere tra gli attributi del pacchetto Accounting-Request l'attributo Acct-Delay-Time il cui valore verrà aggiornato quando il pacchetto viene ritrasmesso, cambiando il contenuto del campo Attributes e richiedendo un nuovo valore per il campo Identifier e un nuovo Request Authenticator.

I pacchetti Accounting-Response sono inviati dal server RADIUS al client per confermare che il pacchetto Accounting-Request è stato ricevuto e registrato con successo. Se l'Accounting-Request è stato registrato con successo allora il server RADIUS deve trasmettere un pacchetto con il campo Code settato a 5 (Accounting-Response). Alla ricezione di un Accounting-Response da parte del client, il campo Identifier viene confrontato con quello del pacchetto Accounting-

Capitolo 4 RADIUS: Pacchetti e Attributi

Request in sospeso. Il campo Response Authenticator deve contenere la corretta risposta per l'Accounting Request in sospeso. Pacchetti non validi vengono silenziosamente scartati.

Ad un pacchetto Accounting-Response non è richiesto avere alcun attributo.

4.3 Attributi

In questo paragrafo si descrivono i valori del campo Attribute che possono essere presenti nei pacchetti analizzati in precedenza.

Ovviamente si analizzeranno solo i valori più importanti lasciando il resto a [1] e [11].

Gli attributi sono trasmessi dentro al pacchetto RADIUS in un formato standard predeterminato come mostrato in figura 4.2

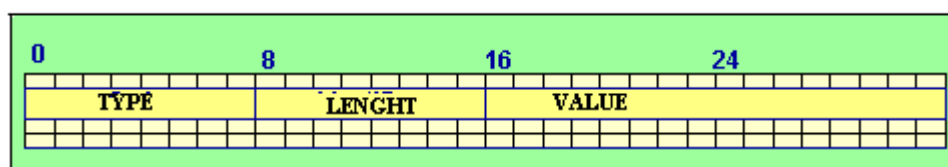


Figura 4.2 Formato standard attributi

- *Type*: composto da un ottetto. questo numero denota il tipo di attributo presente nel pacchetto. Il nome dell'attributo non viene passato nel pacchetto. Generalmente, l'attributo Type ha un range tra 1-255.
- *Length*: composto da un ottetto e indica la lunghezza

Capitolo 4 RADIUS: Pacchetti e Attributi

dell'attributo, compreso i campi Type, Length e Value. Se un attributo viene ricevuto in un pacchetto Access-Accept, Access-Reject o Access-Challenge con una lunghezza non valida, il pacchetto deve essere o trattato come un Access-Reject o può essere scartato.

- *Value*: questo campo può variare da 0 o più ottetti e contiene informazioni specifiche dell'attributo. Il formato e la lunghezza del campo Value è determinata dai campi Length e Type.

Tutti gli attributi devono avere un valore, anche se il valore dell'attributo è nullo. I valori rappresentano l'informazione da trasportare per cui ogni particolare attributo è stato progettato. Essi trasportano il “succo” dell'informazione.

L'implementazione RADIUS, in accordo con l'RFC, è disegnata per cercare certi tipi di valori nel campo Value di un particolare attributo. Per esempio, non dovrebbe essere possibile trovare numeri random in un attributo creato per la data, né potrà essere possibile avere un indirizzo IP in una stringa di caratteri random. Per diminuire la confusione sui diversi valori dell'attributo che vengono passati in una trasmissione, ogni valore corrispondente dell'attributo viene assegnato ad un certo tipo. Questo descrive semplicemente il tipo di valore , se un numero, un indirizzo IP, o una data.

Ci sono 4 tipi delineati in RFC:

- *Integer*: valore di 32 bit senza segno
- *Address*: valore di 32 bit
- *String*: 1-253 ottetti contenenti dati binari. Le stringhe di lunghezza 0 non vengono inviate
- *Time*: valore di 32 bit

Capitolo 4 RADIUS: Pacchetti e Attributi

User-Name

Questo attributo indica il nome dell'utente che deve essere autenticato. Deve essere inviato in un pacchetto Access-Request se disponibile. Il campo Type viene impostato a 1.

Esso può essere inviato in un pacchetto Access-Accept, nel qual caso il client dovrebbe usare il nome inviato nel pacchetto Access-Accept in tutti i pacchetti Accounting-Request per quella sessione. Se l'Access-Accept include il Service-Type= Rlogin e l'attributo User-Name, un NAS può usare l'User-Name restituito quando si compie la funzione Rlogin.

Il campo Type dell'attributo User-Name viene impostato a 1, il campo Length è maggiore di 3. il campo Value è una stringa che può essere di queste tre forme:

3. text
4. l'identificativo dell'accesso alla rete descritto nell'RFC 2486
5. nome distinto: un nome nella forma ASN.1 usato nei sistemi Public Key Authentication

User-Password

questo attributo indica la password dell'utente che deve essere autenticato, o l'input dell'utente subito dopo l'Access-Challenge. Viene utilizzato solo nei pacchetti Access-Request. In questo tipo di attributo il campo Type è impostato a 2.

Durante la trasmissione la password viene nascosta. La password viene prima riempita all'estremità con valori nulli per raggiungere un multiplo di 16 ottetti. Viene calcolata poi una funzione di hash MD5 sugli ottetti indicati prima composti da shared secret seguita dal Request Authenticator. A questo valore

Capitolo 4 RADIUS: Pacchetti e Attributi

viene eseguito lo Xor con il primo segmento dei 16 ottetti della password e posti nei primi 16 ottetti del campo String dell'attributo User-Password.

Se la password è più lunga di 16 caratteri, viene calcolata una seconda funzione di hash MD5 sugli ottetti composti da shared secret seguito dal risultato del primo Xor. A questo hash viene fatto di nuovo lo Xor con i secondi 16 ottetti della password e messi nei secondi 16 ottetti del campo String dell'attributo User-Password.

Il campo Type dell'attributo User-Password è settato a 2, il campo length deve essere compreso tra 18 e 130.

Chap-Password

Questo attributo indica il valore di risposta fornito da un utente PPP- CHAP (challenge Handshake Authentication Protocol) in risposta ad una sfida. È utilizzato solo nei pacchetti Access-Request. Il valore di sfida CHAP si trova nell'attributo CHAP-Challenge (valore campo Type 60) se presente nel pacchetto, altrimenti nel campo Request Authenticator.

Il campo Type di questo attributo è settato a 3.

Attributo NAS-IP-Address

Questo attributo indica l'indirizzo IP del NAS che sta chiedendo l'autenticazione dell'utente. Il NAS-IP-Address viene utilizzato solo nei pacchetti Access-Request. Il campo Type viene settato a 4 .

Attributo Session-Timeout

L'attributo indica il tempo massimo, espresso in secondi, che l'utente può rimanere

Capitolo 4 RADIUS: Pacchetti e Attributi

connesso alla rete prima che il client RADIUS lo butti fuori. Questo attributo è usato principalmente per implementare il limite di tempo di connessione per un pacchetto di tipo account o per prevenire il camping ¹⁶sulla linea.

Questo attributo viene utilizzato nei pacchetti Access-Accept e in quelli Access-Challenge.

Il campo Type dell'attributo viene settato con il valore 27

Attributo Terminate-Action

Questo attributo indica quale azione il NAS dovrebbe prendere quando il collegamento del client termina. È usato solo nei pacchetti Access-Accept.

Ci sono due valori possibili per questo attributo:

4. 0 indica che il server dovrebbe terminare il servizio di default;
5. 1 indica che il client desidera che il NAS invii un nuovo pacchetto Access-REquest una volta completata la sessione.

Il campo Type è settato con il valore 29.

Attributo Acct-Status-Type

Questo attributo indica se l'Accounting-Request segna l'inizio del servizio dell'utente (Start) o la fine (Stop).

Esso può essere usato dal client per indicare l'inizio dell'accounting specificando Accounting-ON e indicare la fine dell'accounting specificando Accounting-OFF.

Il campo Type viene settato a 40 per l'Acct-Status-Type.

¹⁶ Quando un utente tratta una transazione non dedicata, come il dial-up, come una connessione dedicata utilizzando la linea 24 ore al giorno, sette giorni alla settimana

Capitolo 4 RADIUS: Pacchetti e Attributi

Il campo Value è di 4 ottetti e il suo valore indica:

1-Start

2-Stop

7-Accounting-On

8-Accounting-Off

15-Reserved for Failed

Attributo Acct-Delay-Time

Questo attributo indica per quanti secondi il client ha provato a inviare un record.

Cambiando l'attributo Acct-Delay-Time cambia anche l'Identifier.

Il campo Type viene settato a 41

Attributo Acct-Session-Id

Questo attributo è un ID univoco di Accounting per rendere semplice confrontare i record di start e di stop in un file di log. I record di start e stop per una data sessione devono avere lo stesso Acct-Session-ID. Un pacchetto Accounting-Request deve avere un Acct-Session-ID. Un pacchetto Access-Request può avere un Acct-Session-ID; se è così, allora il NAS deve usare lo stesso Acct-Session-ID nei pacchetti Accounting-Request per la stessa sessione.

Il campo type è impostato a 44 per l'Acct-Session-ID

Attributo Acct-Session-Time

Questo attributo indica da quanto tempo l'utente ha ricevuto il servizio, e può solo

Capitolo 4 RADIUS: Pacchetti e Attributi

essere presente nel record Accounting-Request dove Acct-Status-Type è settato a Stop.

Il campo Type è settato a 46 per l'Acct-Session-Time

Attributo Acct-Input-Packets

Questo attributo indica quanti pacchetti sono stati ricevuti su una determinata porta può essere presente solo nei record Accounting-Request dove l'Acct-Status-Type è settato a Stop.

Il valore del campo Type è settato a 47.

Attributo Acct-Output-Packets

Questo attributo indica quanti pacchetti sono stati inviati dalla porta da cui è inviato il servizio. Questo attributo può essere presente solo nei record Accounting-Request dove l'Acct-Status-Type è settato a Stop.

Il valore del campo Type è settato a 48.

4.5 Sicurezza Radius

Per fornire la sicurezza ai messaggi RADIUS, il client e il server RADIUS sono configurati con una la stessa shared secret. La shared secret è usata per mettere al sicuro il traffico RADIUS ed è configurata sia sul client che sul server come stringhe di testo.

Questa sessione descriverà il problema della sicurezza di RADIUS e i possibili attacchi ai server RADIUS. Questi problemi dipendono dall'abilità dell'attacker di catturare i messaggi RADIUS inviati tra il client e il server RADIUS. Questo

Capitolo 4 RADIUS: Pacchetti e Attributi

implica che l'attacker ha l'accesso fisico alla rete ed è possibile per lui inserirsi tra il client e il server RADIUS.

Messaggi Radius Access-Request inviati dal client RADIUS non sono autenticati: di default, non si verifica la crittografia quando arriva un messaggio Access-Request dal server RADIUS. Il server RADIUS verifica solo che il messaggio sia originato da l'indirizzo IP del client che il server ha configurato, ma l'indirizzo IP sorgente per i messaggi RADIUS possono facilmente subire un attacco di tipo spoofing.

La soluzione a questo tipo di attacco è quella di richiedere, da parte del server, l'attributo Message-Authenticator in tutti i pacchetti Access-Request. L'attributo Message-Authenticator è un hash MD5 dell'intero messaggio usando la shared secret come chiave. Il server di accesso deve inviare messaggi con l'attributo Message-Authenticator e il server Radius deve silenziosamente scartare il messaggio se questo attributo non è presente o la sua verifica fallisce. Di solito l'attributo è richiesto solo per i messaggi EAP over RADIUS.

La debolezza della shared secret può essere dovuta alla mediocre configurazione e alla dimensione ridotta: in molte installazioni di RADIUS, la stessa shared secret è usata per proteggere molte coppie client-server, e la shared secret non è sufficientemente casuale per impedire che un dictionary attack offline abbia successo. Per ipotesi della shared secret RADIUS, il campo Respose-Authenticator e il contenuto dell'attributo del Message-Authenticator possono facilmente essere elaborati. Questi risultati sono confrontati con i valori all'interno di un messaggio Access-Accept, Access-Reject o Access-Challenge catturato.

La situazione è peggiore nelle implementazioni del client e del server RADIUS che limitano la dimensione della shared secret e richiedono che essa debba comprendere solo le caratteristiche che possono essere scritte su una tastiera, che usa solo 94 dei possibili 2566 possibili codici ASCII.

La soluzione a questo problema è:

Capitolo 4 RADIUS: Pacchetti e Attributi

- se la shared secret deve essere una sequenza dei caratteri della tastiera, scegliere una shared secret che sia di almeno 22 caratteri e possibilmente che sia una sequenza casuale di lettere maiuscole e minuscole, numeri e punteggiatura. Se la shared secret può essere configurata come una sequenza di cifre esadecimali, usare almeno 32 caratteri esadecimali casuali.
- Usare una diversa shared secret per ogni coppia RADIUS server-RADIUS client.

Attributi sensibili sono cifrati usando il meccanismo per nasconderli di RADIUS: il meccanismo RADIUS di nascondere usa la RADIUS shared secret, il Request-Authenticator, e l'uso dell'algoritmo di hashing MD5 per cifrare lo User-Password e altri attributi come la Tunnel-Password (rfc 2868).

L'uso dello stream cipher e del MD5 come primitive di cifratura sono parte delle specifiche RADIUS. Il solo modo standard per proteggere ulteriormente gli attributi che sono nascosti è l'uso dell'IPsec (Internet Protocol Security) con l'ESP (Encapsulating Security Payload) e un algoritmo di cifratura come il 3DES (Triple Data Encryption Standard), per fornire la confidenzialità dei dati dell'intero messaggio RADIUS. Se queste ultime tre opzioni non sono possibili, le implementazioni RADIUS e gli amministratori di rete possono minimizzare la loro vulnerabilità facendo quanto segue:

- richiedere l'uso dell'attributo Message-Authenticator(RFC 2869) su tutti i messaggi Access-Request.
- Usare la crittografia forte per il Request Authenticator
- richiedere l'uso di password utente forte.
- Usare un meccanismo di conteggio e di bloccaggio dell'autenticazione per prevenire un attacco online di tipo dizionario contro le password dell'utente.

Capitolo 4 RADIUS: Pacchetti e Attributi

- Usare una shared secret di 128 bit.

Valori Request Authenticator scarsi possono essere usati per decifrare gli attributi cifrati: nell'RFC 2865, un Request Authenticator sicuro deve essere temporaneamente e globalmente unico. Il Request Authenticator e la shared secret sono utilizzati insieme per determinare il key stream usato per cifrare la User-Password e altri attributi.

È possibile per un attacker con la possibilità di catturare il traffico tra il client e il server ottenere l'accesso alla rete per creare un dizionario dei Request Authenticators RADIUS e dei corrispondenti key stream usati per cifrare la User-Password e altri attributi. Se il valore del Request Authenticator viene sempre ripetuto da un client usando la stessa shared secret, allora la User-Password e gli altri attributi contenuti all'interno dei pacchetti scambiati possono essere determinati.

Se il Request Authenticator non è abbastanza casuale, allora esso può essere predetto e molto probabilmente, anche, ripetuto. Il generatore del Request Authenticator dovrebbe avere la caratteristica di essere crittografato. Se così non fosse, è possibile usare Ipsec con ESP e un algoritmo di crittografia come il 3DES per fornire la confidenzialità dei dati per l'intero messaggio RADIUS come descritto nell'RFC 3162.

Capitolo 5 FreeRadius

5.1 Introduzione

Freeradius è una delle principali implementazioni OpenSource del server RADIUS. Il software è rilasciato sotto licenza GNU/GPL ed è liberamente utilizzabile ed è possibile reperirlo in [12].

Questo software è basato sull'implementazione del server RADIUS di Livingston, ma è stato completamente riscritto. Tra le features offerte sono disponibili:

- la possibilità di un sistema di configurazione via web
- la possibilità di limitare il numero massimo di connessioni simultanee
- il supporto di più metodi di autenticazione
- permettere o negare l'accesso a gruppi di utenti
- il file di configurazione è modulare, è possibile inserire le sezioni con il comando INCLUDE
- supporta le specifiche dei più importanti produttori tra cui 3com/USR, Cisco.

Freeradius offre il supporto per LDAP, MYSQL, Oracle. I tipi di autenticazione gestiti da questo software sono EAP con i diversi moduli EAP-MD5, EAP-TLS, EAP-TTLS e EAP-PEAP, MSCHAP, MSCHAPv2, Cisco LEAP oltre ai classici PAP e CHAP

5.2 Installazione FreeRadius

L'ambiente di sviluppo del FreeRadius è sulla distribuzione Linux Fedora Core 5. Come nella maggior parte delle distribuzioni, anche nel nostro ambiente, ci sono dei pacchetti precompilati per l'installazione di FreeRadius, basta digitare il comando:

```
# yum install freeradius
```

Comunque è possibile installare FreeRadius scaricandolo dal sito ufficiale. In tal caso l'installazione di base prevede diversi passaggi:

```
# tar xfvz freeradius.x.y.z.tar.gz
```

```
# cd freeradius x.y.z
```

```
# ./configure
```

```
# make
```

```
# make install
```

5.3 File di configurazione di FreeRadius

FreeRadius è composto da diversi file di configurazione e di log, i quali dovrebbero trovarsi nel percorso `/etc/radb`, ma questo percorso varia da distribuzione a distribuzione.

La configurazione di FreeRadius consiste nell'impostazione del server, dei client e degli utenti.

Questi componenti sono gestiti dai seguenti file di configurazione:

Capitolo 5 FreeRadius

radius.conf

clients.conf

user

sql.conf nel caso in cui gli utenti vengono gestiti da database mysql

5.3.1 radiusd.conf

Il file *radiusd.conf* è la sede centrale di molti aspetti di configurazione di FreeRadius. Esso include le direttive di configurazione. Ci sono anche le opzioni di configurazione generale per i diversi moduli disponibili. I moduli possono richiedere opzioni generiche, e FreeRadius passerà queste opzioni definite al modulo attraverso le sue API.

Questo file di configurazione è molto grande, ma solo alcune sezioni servono per ottenere un funzionamento decoroso.

La struttura principale dei file radiusd.conf è la seguente:

```
[opzioni globali]
modules{
}
authorize{
}
authenticate{
}
accounting{
}
```

Capitolo 5 FreeRadius

All'inizio sono definite le opzioni globali, che specificano ad esempio i percorsi per le varie cartelle ed i parametri di funzionamento del server, impostazioni che tipicamente si possono lasciare invariate. Nella prima sezione `modules` sono definiti i singoli moduli che si possono usare indistintamente per autorizzazione, autenticazione e accounting, mentre le sezioni `authorize`, `authenticate` e `accounting` stabiliscono quali moduli sono necessari per ciascuna operazione.

Nella sezione `modules` sono disponibili molti moduli. A seconda del contesto di implementazione di FreeRadius possiamo utilizzare:

```
modules{
    pap{
        ..... }
    chap{
        ..... }
    ldap{
        ..... }
}
```

Alcune sezioni possono essere inserite nel file di configurazione grazie al comando `INCLUDE`, grazie alla già citata modularità di FreeRadius. Questi file possono riguardare l'autenticazione EAP e quella attraverso il database sql.

```
$INCLUDE ${confdir}/eap.conf
```

```
$INCLUDE ${confdir}/sql.conf
```

Le sezioni `authorize`, `authenticate`, `accounting` contengono i nomi dei moduli precedentemente configurati

5.3.2 File di configurazione clients.conf

Questo file contiene le informazioni relative ai client ovvero ai nodi che svolgono il ruolo di NAS. In passato queste si potevano inserire nel file *clients*, ma l'utilizzo odierno di quest'ultimo è deprecato.

Di seguito viene riportato il formato delle direttive presente nel file *clients.conf*

```
client indirizzo_ip | hostname {  
    secret = chiave_privata  
    shortname = nome_del_client_per_i_log  
    nastype = tipo_di_nas }
```

dove:

client: definisce il NAS tramite un indirizzo Ip oppure un hostname.

secret: definisce la password in plaintext che i client dovranno utilizzare per connettersi al server RADIUS quando richiedono l'autenticazione di un supplicant. La lunghezza massima per il valore di questa chiave privata è di 31 caratteri.

shortname: viene utilizzato come alias da sostituire all'hostname o all'indirizzo Ip ed è adoperato nei log. La sua specifica è obbligatoria.

nastype: specifica il tipo di NAS. Opzioni valide sono: *cisco*, *computone*, *livingston*, *max40xx*, *multitech*, *netserver*, *pathras*, *patton*, *portslave*, *tc*, *usrhiper* ed *other*.

5.3.3 Il file di configurazione users

Questo file contiene informazioni di configurazione che stabiliscono quali utenti possono autenticarsi e quale protocollo impiegare per questa procedura. Ogni direttiva può assumere una forma piuttosto complessa, perché le possibilità offerte da FreeRADIUS sono ampie, tuttavia nella maggioranza dei casi la forma impiegata è molto semplice:

```
nome_utente      Auth-Type = tipo di autenticazione
```

dove:

nome_utente: specifica il nome dell'utente e non può superare la lunghezza massima di 253 caratteri.

Auth_Type: definisce il protocollo di autenticazione.

Quando il server riceve una richiesta da parte di un NAS scruta il file dall'inizio sin quando non trova una direttiva che specifica come nome utente una stringa che combacia con quella riportata nel pacchetto. In caso positivo è applicato il protocollo di autenticazione definito nella dichiarazione. E' importante sottolineare che solamente la prima direttiva che combacia viene utilizzata per l'autenticazione, a meno che non sia presente l'opzione *Fall-Through*, la quale impone, in caso di insuccesso, di esaminare anche quelle successive.

Il problema di questa formulazione consiste nello specificare lo *user_name* per ogni utente e questa procedura non è affatto comoda. Per questa ragione tale parametro può essere sostituito dalla stringa "*DEFAULT*", la quale consente di costruire direttive applicabili in ogni caso, vale a dire qualunque sia lo *user_name*.

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

In questo progetto è stata impiegata la distribuzione linux Fedora Core 7.

Per l'installazione del sistema si sono utilizzati pacchetti precompilati con il comando yum, installando il pacchetto freeradius con il pacchetto mysql di cui abbiamo bisogno per sviluppare la nostra applicazione.

Infatti freeradius sarà configurato per cercare le informazioni di autenticazione sul database mysql.

I file di configurazione di freeradius sono contenuti nella directory `/etc/raddb/`.

La struttura che si andrà ad implementare viene mostrata nella figura seguente:



Appendice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

Con il comando

```
# rpm -qa freeradius freeradius-mysql
```

controlliamo che il pacchetto freeradius sia installato e che ci sono le librerie per far comunicare freeradius con mysql.

Con il comando

```
# rpm -qa mysql
```

controlliamo che il pacchetto mysql sia installato sul nostro sistema.

Il nostro server radius apparterrà alla rete LAN a cui appartiene l'Access Point.

Gli indirizzi saranno entrambi statici.

A.1 Configurazione utenti

Gli utenti vengono autenticati grazie al database mysql. Per far dialogare mysql con freeradius dobbiamo prima di tutto configurare mysql creando un database dove inserire gli utenti.

Configurazione mysql

Impostare la password dell'utente amministratore del database:

```
$ mysqladmin -u root password 'secret'
```

creare poi un database all'interno di mysql

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

```
$ mysql -u root -p
```

```
mysql> CREATE DATABASE radius;
```

```
mysql> quit
```

Creare la struttura del database ed un utente con i privilegi di accesso e modifica del database radius:

```
cat /usr/share/doc/freeradius/examples/mysql.sql | mysql -u root -p radius
```

```
mysql -u root -p
```

```
mysql> GRANT ALL PRIVILEGES ON radius.* TO 'radius'@'localhost'  
IDENTIFIED BY 'secret';
```

```
mysql> FLUSH PRIVILEGES;
```

```
mysql> quit
```

Quindi inseriamo un utente nel database radius:

```
mysql>use radius;
```

```
mysql>INSERT INTO radcheck (UserName, Attribute, Value) VALUES  
( 'mysqltest', 'Password', 'testsecret');
```

Appendice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

A questo punto avremo la tabella radcheck con il seguente record:

```
mysql> select * from radcheck;
```

```
+----+-----+-----+----+-----+
| id | Username | Attribute | op | Value      |
+----+-----+-----+----+-----+
| 1  | mysqltest | Password  | == | testsecret |
+----+-----+-----+----+-----+
```

A.2 Configurazione di freeradius

Freeradius è configurato per autenticare gli utenti usando l'autenticazione MSCHAPv2. I dati di accesso verranno prelevati dal database mysql.

Per l'autenticazione MSCHAPv2 dovrà essere incluso nel file di configurazione radiusd.conf la sezione riguardante il file eap.conf.

In questo file è stata impostata come tipo di autenticazione paep, la quale utilizza i certificati TLS per poter funzionare.

Nel file sql.conf, anch'esso da includere nel file di configurazione radiusd.conf, è stato inserito il nome utente dell'amministratore del database mysql e la relativa password; infine è stato delineato quale database freeradius deve utilizzare per autenticare gli utenti.

Vediamo adesso la configurazione dei file radiusd.conf e eap.conf e sql.conf

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

Il file /etc/raddb/radiusd.conf

```
prefix = /usr
exec_prefix = /usr
sysconffdir = /etc
localstatedir = /var
sbindir = /usr/sbin
logdir = ${localstatedir}/log/radius
raddbdir = ${sysconffdir}/raddb
radacctdir = ${logdir}/radacct
confdir = ${raddbdir}
run_dir = ${localstatedir}/run/radiusd
log_file = ${logdir}/radius.log
libdir = /usr/lib
pidfile = ${run_dir}/radiusd.pid
user = radiusd
group = radiusd
max_request_time = 30
delete_blocked_requests = no
cleanup_delay = 5
max_requests = 1024
bind_address = *
port = 0
hostname_lookups = no
allow_core_dumps = no
regular_expressions = yes
extended_expressions = yes
log_stripped_names = no
```

Appendice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

```
log_auth = yes
log_auth_badpass = yes
log_auth_goodpass = yes
usercollide = no
lower_user = no
lower_pass = no
nospace_user = no
nospace_pass = no
checkrad = ${sbindir}/checkrad

security {

    max_attributes = 200
    reject_delay = 1
    status_server = no
}

proxy_requests = no
$INCLUDE ${confdir}/clients.conf
snmp = no
thread pool {
    start_servers = 5
    max_servers = 32
    min_spare_servers = 3
    max_spare_servers = 10
    max_requests_per_server = 0
}
modules {
```

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

```
$INCLUDE ${confdir}/eap.conf
    realm suffix {
        format = suffix
        delimiter = "@"
        ignore_default = no
        ignore_null = no
    }
    realm realmpercent {
        format = suffix
        delimiter = "%"
        ignore_default = no
        ignore_null = no
    }
    realm ntdomain {
        format = prefix
        delimiter = "\\"
        ignore_default = no
        ignore_null = no
    }
    preprocess {
        huntgroups = ${confdir}/huntgroups
        hints = ${confdir}/hints
        ascend_channels_per_line = 23
        with_ntdomain_hack = no
        with_specialix_jetstream_hack = no
        with_cisco_vsa_hack = no
    }
    files {
```

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

```
usersfile = ${confdir}/users
acctusersfile = ${confdir}/acct_users
preproxy_usersfile = ${confdir}/preproxy_users
compat = no
}
detail {
    detailfile = ${radacctdir}/%{Client-IP-Address}/detail-%Y%m%d
    detailperm = 0600
}
$INCLUDE ${confdir}/sql.conf
radutmp {

    filename = ${logdir}/radutmp
    username = %{User-Name}
    case_sensitive = yes
    check_with_nas = yes
    perm = 0600
    callerid = "yes"
}

radutmp sradutmp {
    filename = ${logdir}/sradutmp
    perm = 0644
    callerid = "no"
}
attr_filter {
    attrsfile = ${confdir}/attrs
}
counter daily {
```

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

```
filename = ${raddbdir}/db.daily
key = User-Name
count-attribute = Acct-Session-Time
reset = daily
counter-name = Daily-Session-Time
check-name = Max-Daily-Session
allowed-servicetype = Framed-User
cache-size = 5000
}
always fail {
    rcode = fail
}
always reject {
    rcode = reject
}
always ok {
    rcode = ok
    simulcount = 0
    mpp = no
}
expr {
}
digest {
}
exec {
    wait = yes
    input_pairs = request
}
exec echo {
```

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

```
        wait = yes
        program = "/bin/echo %{User-Name}"
        input_pairs = request
        output_pairs = reply
    }
    ippool main_pool {
        range-start = 192.168.1.1
        range-stop = 192.168.3.254
        netmask = 255.255.255.0
        cache-size = 800
        session-db = ${raddbdir}/db.ippool
        ip-index = ${raddbdir}/db.ipindex
        override = no
        maximum-timeout = 0
    }
}
instantiate {
    exec
    expr
}
authorize {
    preprocess
    auth_log
    mschap
    suffix
    eap
    files
    sql
}
```

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

```
authenticate {
    Auth-Type MS-CHAP {
        mschap
    }
    unix
    eap
}
preacct {
    preprocess
    acct_unique
    suffix
#   files
}
accounting {
    detail
    unix
    radutmp
    sql
}
session {
    radutmp
    sql
}
post-auth {
}
pre-proxy {
}
post-proxy {
}
```

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

File di eap.conf

```
eap {
    default_eap_type = peap
    timer_expire     = 60
    ignore_unknown_eap_types = no
    cisco_accounting_username_bug = no
    md5 {
    }
    leap {
    }
    gtc {
    }
    tls {
        private_key_password = whatever
    private_key_file = ${raddbdir}/certs/cert-srv.pem
    certificate_file = ${raddbdir}/certs/cert-srv.pem
    CA_file = ${raddbdir}/certs/demoCA/cacert.pem
        dh_file = ${raddbdir}/certs/dh
        random_file = ${raddbdir}/certs/random
        fragment_size = 1024
        include_length = yes
        check_crl = yes
    }
    peap {
        default_eap_type = mschapv2
    }
    mschapv2 {
    }
}
```


Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

File sql.conf

```
sql {  
    driver = "rlm_sql_mysql"  
  
    server = "localhost"  
    login = "root"  
    password = "secret"  
  
    radius_db = "radius"  
  
    acct_table1 = "radacct"  
    acct_table2 = "radacct"  
  
    postauth_table = "radpostauth"  
  
    authcheck_table = "radcheck"  
    authreply_table = "radreply"  
  
    groupcheck_table = "radgroupcheck"  
    groupreply_table = "radgroupreply"  
  
    usergroup_table = "usergroup"  
  
    nas_table = "nas"  
  
    deletestalesessions = yes
```

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

sqltrace = no

sqltracefile = \${logdir}/sqltrace.sql

num_sql_socks = 5

connect_failure_retry_delay = 60

sql_user_name = "%{User-Name}"

```
authorize_check_query = "SELECT id, UserName, Attribute, Value, op \
FROM ${authcheck_table} \
WHERE Username = '%{SQL-User-Name}' \
ORDER BY id"
```

```
authorize_reply_query = "SELECT id, UserName, Attribute, Value, op \
FROM ${authreply_table} \
WHERE Username = '%{SQL-User-Name}' \
ORDER BY id"
```

```
authorize_group_check_query = "SELECT
${groupcheck_table}.id,${groupcheck_table}.GroupName,${groupcheck_table}.
Attribute,${groupcheck_table}.Value,${groupcheck_table}.op FROM
${groupcheck_table},${usergroup_table} WHERE ${usergroup_table}.Username
= '%{SQL-User-Name}' AND ${usergroup_table}.GroupName =
${groupcheck_table}.GroupName ORDER BY ${groupcheck_table}.id"
```

```
simul_verify_query = "SELECT RadAcctId, AcctSessionId, UserName,
NASIPAddress, NASPortId, FramedIPAddress, CallingStationId, FramedProtocol
FROM ${acct_table1} WHERE UserName='%{SQL-User-Name}' AND
AcctStopTime = 0"
```

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

```
group_membership_query = "SELECT GroupName FROM
${usergroup_table} WHERE UserName='%{SQL-User-Name}'"
```

```
postauth_query = "INSERT into ${postauth_table} (id, user, pass, reply,
date) values ('', '%{User-Name}', '%{User-Password:-Chap-Password}',
'%{reply:Packet-Type}', NOW())"
```

```
}
```

File clients.conf

```
client 192.168.1.252 {
    secret = testing123
    shortname = A.P.
}
```

Configurazione Access Point

L'access point, al quale si conetterà il client wireless per accedere alla rete e avere la connessione ad internet, verrà configurato come mostrato nella figura A.1

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

The screenshot displays the configuration interface for an Access Point, specifically the RADIUS settings. The interface is organized into several sections separated by horizontal lines. At the top, the 'Security Mode' is set to 'RADIUS' via a dropdown menu. Below this, the 'RADIUS Server' is configured with IP address 192.168.1.240 and 'RADIUS Port' 1812. The 'Shared Secret' is set to 'testing123'. The 'Encryption' is set to '40/64-bit(10 hex digits)'. A 'Passphrase' of 'laratux' is entered, with a 'Generate' button to its right. Below the passphrase, four keys are listed: Key 1 (A64C1AE494), Key 2 (490F79B872), Key 3 (123E4190CF), and Key 4 (FE5F5E2BD9). The 'TX Key' is set to 1. At the bottom of the form, there are two buttons: 'Save Settings' and 'Cancel Changes'.

Security Mode:	RADIUS	
RADIUS Server:	192 . 168 . 1 . 240	
RADIUS Port:	1812	
Shared Secret:	testing123	
Encryption:	40/64-bit(10 hex digits)	
Passphrase:	laratux	Generate
Key 1:	A64C1AE494	
Key 2:	490F79B872	
Key 3:	123E4190CF	
Key 4:	FE5F5E2BD9	
TX Key:	1	

Fig A.1 Configurazione Access Point

Nella sezione WIRELESS dell'interfaccia grafica dell'Access Point, vediamo che esso viene collegato al server Radius impostando alla voce Security Mode la voce RADIUS e di seguito impostare l'indirizzo IP del server, la porta su cui avviene l'autenticazione e la shared secret da condividere con il server RADIUS

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

A.3 LOG di connessione all'Access Point con autenticazione MSCHAPv2

rad_recv: Access-Request packet from host 192.168.1.252:3072, id=0, length=173

```
User-Name = "mysqltest"  
NAS-IP-Address = 192.168.1.252  
Called-Station-Id = "001a70323981"  
Calling-Station-Id = "00150044ff48"  
NAS-Identifier = "001a70323981"  
NAS-Port = 51  
Framed-MTU = 1400  
State = 0xc4534d7a539a0d9836fa85dc327de80a  
NAS-Port-Type = Wireless-802.11
```

```
EAP-Message  
0x020800261900170301001b2febac60dac86c25f6fe74f4c62e97dc7591b1a23d044bbafaa21e =
```

```
Message-Authenticator = 0x3914d5c65a5ba0c3645d5e6137bf615d
```

```
Processing the authorize section of radiusd.conf  
modcall: entering group authorize for request 8  
modcall[authorize]: module "preprocess" returns ok for request 8  
modcall[authorize]: module "chap" returns noop for request 8  
modcall[authorize]: module "mschap" returns noop for request 8  
  rlm_realm: No '@' in User-Name = "mysqltest", looking up realm NULL  
  rlm_realm: No such realm "NULL"  
modcall[authorize]: module "suffix" returns noop for request 8  
rlm_eap: EAP packet type response id 8 length 38  
rlm_eap: No EAP Start, assuming it's an on-going EAP conversation  
modcall[authorize]: module "eap" returns updated for request 8  
radius_xlat: 'mysqltest'  
  
rlm_sql (sql): sql_set_user escaped user --> 'mysqltest'  
radius_xlat: 'SELECT id, UserName, Attribute, Value, op      FROM radcheck      WHERE  
Username = 'mysqltest'      ORDER BY id'  
  
rlm_sql (sql): Reserving sql socket id: 2  
  
radius_xlat:'SELECT  
radgroupcheck.id,radgroupcheck.GroupName,radgroupcheck.Attribute,radgroupcheck.Value,radgr  
oupcheck.op FROM radgroupcheck,usergroup WHERE usergroup.Username = 'mysqltest' AND  
usergroup.GroupName = radgroupcheck.GroupName ORDER BY radgroupcheck.id'  
  
radius_xlat: 'SELECT id, UserName, Attribute, Value, op      FROM radreply      WHERE  
Username = 'mysqltest'      ORDER BY id'  
  
radius_xlat:'SELECT
```

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

```
radgroupreply.id,radgroupreply.GroupName,radgroupreply.Attribute,radgroupreply.Value,radgroupreply.op FROM radgroupreply,usergroup WHERE usergroup.Username = 'mysqltest' AND usergroup.GroupName = radgroupreply.GroupName ORDER BY radgroupreply.id'
```

```
rlm_sql (sql): Released sql socket id: 2
  modcall[authorize]: module "sql" returns ok for request 8
rlm_pap: Found existing Auth-Type, not changing it.
  modcall[authorize]: module "pap" returns noop for request 8
modcall: leaving group authorize (returns updated) for request 8
  rad_check_password: Found Auth-Type EAP
auth: type "EAP"
  Processing the authenticate section of radiusd.conf
modcall: entering group authenticate for request 8
  rlm_eap: Request found, released from the list
  rlm_eap: EAP/peap
  rlm_eap: processing type peap
  rlm_eap_peap: Authenticate
  rlm_eap_tls: processing TLS
  eaptls_verify returned 7
  rlm_eap_tls: Done initial handshake
  eaptls_process returned 7
  rlm_eap_peap: EAPTLS_OK
  rlm_eap_peap: Session established. Decoding tunneled attributes.
  rlm_eap_peap: Received EAP-TLV response.
  rlm_eap_peap: Tunneled data is valid.
  rlm_eap_peap: Success
  rlm_eap: Freeing handler
  modcall[authenticate]: module "eap" returns ok for request 8
modcall: leaving group authenticate (returns ok) for request 8
  Processing the post-auth section of radiusd.conf
modcall: entering group post-auth for request 8
  rlm_sql (sql): Processing sql_postauth
```

```
radius_xlat: 'mysqltest'
rlm_sql (sql): sql_set_user escaped user --> 'mysqltest'
radius_xlat: 'INSERT into radpostauth (id, user, pass, reply, date) values ("', 'mysqltest', 'Chap-Password', 'Access-Accept', NOW())'
```

```
rlm_sql (sql) in sql_postauth: query is INSERT into radpostauth (id, user, pass, reply, date) values ("', 'mysqltest', 'Chap-Password', 'Access-Accept', NOW())
rlm_sql (sql): Reserving sql socket id: 1
rlm_sql (sql): Released sql socket id: 1
  modcall[post-auth]: module "sql" returns ok for request 8
modcall: leaving group post-auth (returns ok) for request 8
```

```
Sending Access-Accept of id 0 to 192.168.1.252 port 3072
MS-MPPE-Recv-Key =
0x57cee7769fe9c001d43dca651ba0c32412a959bec5a59900c69542ca2c3e7455
  MS-MPPE-Send-Key =
```

Appedice A Sistema di autenticazione freeradius + mysql + eap/peap + Access Point

```
0xcb11e74433051d6bc1a81a4b079c62f34f4ea2954e2a86a28be10ee88f49cfc4
  EAP-Message = 0x03080004
  Message-Authenticator = 0x00000000000000000000000000000000
  User-Name = "mysqltest"
```

Appendice B Esempio autenticazione captive portal: m0n0wall

La tecnica del captive portal forza un client http a visitare una speciale pagina web, di solito utilizzata per l'autenticazione, prima di poter accedere alla navigazione.

Per far ciò si intercettano tutti i pacchetti che l'utente invia fin dal momento in cui viene aperto il browser e tenta l'accesso su internet. In quel momento il browser viene rediretto verso una pagina web la quale può richiedere l'autenticazione.

Il software captive portal è possibile trovarlo in uso presso gli hotspot Wi-Fi, ma può essere usato per controllare accessi su reti cablate.

Il software utilizzato in questa implementazione è m0n0wall, che permette l'autenticazione attraverso Radius.

Il sistema verrà sviluppato attraverso macchine virtuali dove verrà installato il software m0n0wall e un sistema operativo windows nello specifico windows98.

La macchina virtuale che farà da user è configurata con una scheda di rete per la LAN. La macchina virtuale m0n0wall è stata configurata con due schede di rete: una per la LAN, a cui verrà assegnato l'indirizzo di rete della stessa classe di quello dell'utente e uno per la WAN, il cui indirizzo apparterrà alla stessa classe dell'indirizzo assegnato alla macchina reale su cui è stato configurato FreeRadius.

Nella parte LAN di m0n0wall, durante la configurazione, viene attivato il DHCP, in modo da poter assegnare un indirizzo IP all'host dell'utente.

Appendice B Esempio autenticazione captive portal: m0n0wall

B.1 Configurazione di m0n0wall

Nella sezione INTERFACES configurare gli indirizzi delle interfacce LAN e WAN.

Interfaccia LAN

L'interfaccia LAN viene configurata con un indirizzo IP statico: 192.168.69.1/24

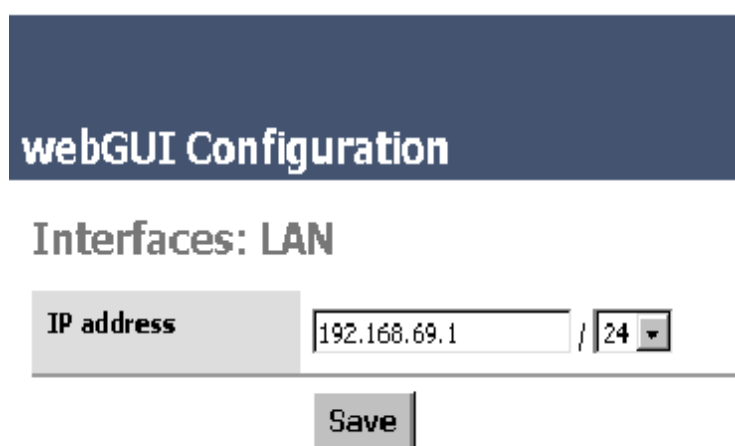


fig. B.1 Configurazione interfaccia LAN

Interfaccia WAN

L'interfaccia WAN viene configurata con un indirizzo IP statico: 192.168.1.50/24 che appartiene alla stessa rete del server RADIUS, fig B.2.

Appendice B Esempio autenticazione captive portal: m0n0wall

webGUI Configuration

Interfaces: WAN

Type

Static IP configuration

IP address	<input type="text" value="192.168.1.50"/> / <input type="text" value="24"/>
Gateway	<input type="text" value="192.168.1.254"/>

Fig. B.2 Configurazione interfaccia WAN

Nella sezione SERVICES selezionare il servizio Captive Portal e configurare l'interfaccia sulla quale il captive portal agirà, fig B.3.

Services: Captive portal

Captive Portal Pass-through MAC Allowed IP addresses Users File n

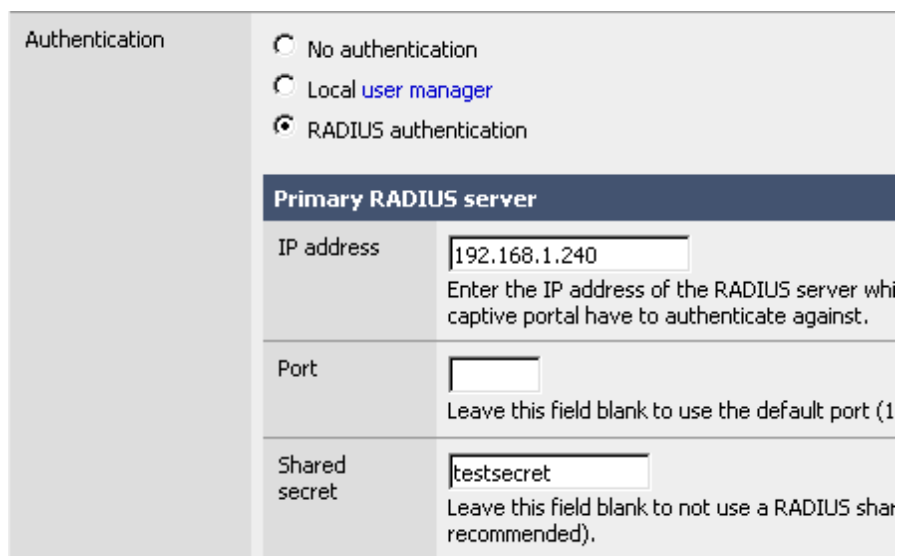
Enable captive portal

Interface
Choose which interface to run the captive portal on.

Fig. B.3 Attivazione del Captive Portal

Appendice B Esempio autenticazione captive portal: m0n0wall

Per attivare l'autenticazione portarsi nella sezione Authenticator e spuntare l'opzione RADIUS Authentication, settare l'indirizzo del server RADIUS e impostare la shared secret che il server userà per dialogare con m0n0wall che agirà come NAS, fig B.4.



The screenshot shows the 'Authentication' section of the m0n0wall configuration interface. It features three radio button options: 'No authentication', 'Local user manager', and 'RADIUS authentication', with the latter being selected. Below these options is a section titled 'Primary RADIUS server' which contains three input fields: 'IP address' (containing '192.168.1.240'), 'Port' (empty), and 'Shared secret' (containing 'testsecret'). Each input field has a descriptive tooltip below it.

Primary RADIUS server	
IP address	<input type="text" value="192.168.1.240"/> Enter the IP address of the RADIUS server which captive portal have to authenticate against.
Port	<input type="text"/> Leave this field blank to use the default port (1
Shared secret	<input type="text" value="testsecret"/> Leave this field blank to not use a RADIUS shared secret (recommended).

fig B.4 Autenticazione tramite RADIUS

Tra le opzioni di RADIUS c'è la possibilità di impostare l'attributo Session-Timeout che fa in modo di disconnettere l'utente dopo un certo periodo di tempo. Per utilizzare questo attributo bisogna spuntare l'opzione Use Session-Timeout Attributes nella sezione RADIUS Options, fig B.5.

Appendice B Esempio autenticazione captive portal: m0n0wall

RADIUS options	
Session-Timeout	<input checked="" type="checkbox"/> Use RADIUS Session-Timeout attributes When this is enabled, clients will be disconnected after the amount of time retrieved from the RADIUS Session-Timeout attribute.
Type	<input type="text" value="default"/> If RADIUS type is set to Cisco, in RADIUS requests (Authentication/Accounting) the value of Calling-Station-Id will be set to the client's IP address and the Called-Station-Id to the client's MAC address. Default behaviour is Calling-Station-Id = client's MAC address and Called-Station-Id = m0n0wall's WAN MAC address.
MAC address format	<input type="text" value="default"/> This option changes the MAC address format used in the whole RADIUS system. Change this if you also need to change the username format for RADIUS MAC authentication. default: 00:11:22:33:44:55 singledash: 001122-334455 ietf: 00-11-22-33-44-55 cisco: 0011.2233.4455 unformatted: 001122334455

Fig. B.5 Flag per impostare l'opzione Timeout

L'ultima opzione necessaria per utilizzare un captive portal è quella di settare una pagina HTML di autenticazione dove l'utente inserirà lo User-Name e la Password, come mostrato nella figura B.6.

Appendice B Esempio autenticazione captive portal: m0n0wall

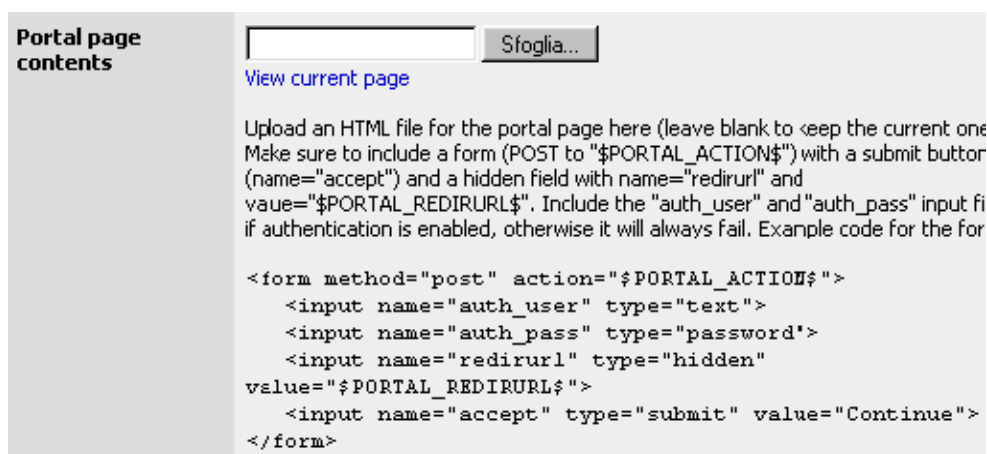


Fig. B.6 Opzione per creare la pagina html di autenticazione

B.2 Configurazione Utenti

Anche nel caso del captive portal gli utenti vengono autenticati attraverso il database mysql.

Per far ciò, dobbiamo inserire l'utente(a), associarlo ad un gruppo(b), associare al gruppo un tipo di autenticazione attraverso l'attributo Auth-Type(c) e, infine, assegnare l'attributo Session-Timeout che farà disconnettere l'utente e attraverso l'attributo Termination-Action richiedere di nuovo l'autenticazione radius settando il valore con RADIUS-Request(d).

Appendice B Esempio autenticazione captive portal: m0n0wall

Tabelle Mysql

(a) mysql> select * from radcheck;

```
+-----+-----+-----+-----+-----+
| id |  UserName  | Attribute  | op | Value      |
+-----+-----+-----+-----+-----+
|  1 | mysqltest  | Password   | == | testsecret |
+-----+-----+-----+-----+-----+
```

(b) mysql> select * from usergroup;

```
+-----+-----+-----+
|  UserName  | GroupName | priority |
+-----+-----+-----+
| mysqltest | default   |         0 |
+-----+-----+-----+
```

(c)mysql> select * from radgroupcheck;

```
+-----+-----+-----+-----+-----+
| id | GroupName      | Attribute  | op | Value      |
+-----+-----+-----+-----+-----+
| 1  | default        | Auth-Type  | := | pap        |
+-----+-----+-----+-----+-----+
```

(d)mysql> select * from radgroupreply;

```
+-----+-----+-----+-----+-----+
| id | GroupName | Attribute          | op | Value      |
+-----+-----+-----+-----+-----+
|  1 | default  | Session-Timeout   | := | 1          |
|  2 | default  | Termination-Action| := | RADIUS-Request |
+-----+-----+-----+-----+-----+
```

Appendice B Esempio autenticazione captive portal: m0n0wall

B.3 Configurazione di freeradius

File clients.conf

In questo esempio il captive portal avrà il ruolo di NAS, quindi il file clients.conf conterrà l'indirizzo dell'interfaccia WAN del captive portal e la stessa shared secret impostata nella sezione Authenticator del captive portal.

Nel nostro caso il file sarà configurato nel seguente modo:

```
client 192.168.1.50 {  
    secret      = testsecret  
    shortname   = mono  
}
```

I file radiusd.conf, sql.conf conservano le configurazioni settate in precedenza.

Appendice B Esempio autenticazione captive portal: m0n0wall

B.4 File di log dell'autenticazione attraverso il captive portal

rad_recv: Access-Request packet from host 192.168.1.50:2517, id=51, length=133

```
NAS-IP-Address = 192.168.1.50
NAS-Identifier = "monowall.local"
User-Name = "mysqltest"
User-Password = "testsecret"
Service-Type = Login-User
NAS-Port-Type = Ethernet
NAS-Port = 0
Framed-IP-Address = 192.168.69.200
Called-Station-Id = "00:0c:29:f0:70:c6"
Calling-Station-Id = "00:0c:29:c8:a5:a7"
```

```
Processing the authorize section of radiusd.conf
modcall: entering group authorize for request 0
modcall[authorize]: module "preprocess" returns ok for request 0
modcall[authorize]: module "chap" returns noop for request 0
modcall[authorize]: module "mschap" returns noop for request 0
  rlm_realm: No '@' in User-Name = "mysqltest", looking up realm NULL
  rlm_realm: No such realm "NULL"
modcall[authorize]: module "suffix" returns noop for request 0
  rlm_eap: No EAP-Message, not doing EAP
modcall[authorize]: module "eap" returns noop for request 0
radius_xlat: 'mysqltest'
```

```
rlm_sql (sql): sql_set_user escaped user --> 'mysqltest'
```

```
radius_xlat: 'SELECT id, UserName, Attribute, Value, op      FROM radcheck      WHERE
Username = 'mysqltest'      ORDER BY id'
rlm_sql (sql): Reserving sql socket id: 4
```

```
radius_xlat:'SELECT
radgroupcheck.id,radgroupcheck.GroupName,radgroupcheck.Attribute,radgroupcheck.Value,radgr
oupcheck.op FROM radgroupcheck,usergroup WHERE usergroup.Username = 'mysqltest' AND
usergroup.GroupName = radgroupcheck.GroupName ORDER BY radgroupcheck.id'
```

```
radius_xlat: 'SELECT id, UserName, Attribute, Value, op      FROM radreply      WHERE
Username = 'mysqltest'      ORDER BY id'
```

```
radius_xlat: 'SELECT
radgroupreply.id,radgroupreply.GroupName,radgroupreply.Attribute,radgroupreply.Value,radgrou
preply.op FROM radgroupreply,usergroup WHERE usergroup.Username = 'mysqltest' AND
usergroup.GroupName = radgroupreply.GroupName ORDER BY radgroupreply.id'
```

```
rlm_sql (sql): Released sql socket id: 4
```


Appendice B Esempio autenticazione captive portal: m0n0wall

```
modcall[authorize]: module "sql" returns ok for request 0
rlm_pap: Found existing Auth-Type, not changing it.
modcall[authorize]: module "pap" returns noop for request 0
modcall: leaving group authorize (returns ok) for request 0
rad_check_password: Found Auth-Type Pap
auth: type "PAP"
Processing the authenticate section of radiusd.conf
modcall: entering group PAP for request 0
rlm_pap: login attempt with password testsecret
rlm_pap: Using clear text password "testsecret".
rlm_pap: User authenticated successfully
modcall[authenticate]: module "pap" returns ok for request 0
modcall: leaving group PAP (returns ok) for request 0
Processing the post-auth section of radiusd.conf
modcall: entering group post-auth for request 0
rlm_sql (sql): Processing sql_postauth
radius_xlat: 'mysqltest'
rlm_sql (sql): sql_set_user escaped user --> 'mysqltest'

radius_xlat: 'INSERT into radpostauth (id, user, pass, reply, date) values ("', 'mysqltest',
'testsecret', 'Access-Accept', NOW())'
rlm_sql (sql) in sql_postauth: query is INSERT into radpostauth (id, user, pass, reply, date) values
("', 'mysqltest', 'testsecret', 'Access-Accept', NOW())

rlm_sql (sql): Reserving sql socket id: 3
rlm_sql (sql): Released sql socket id: 3
modcall[post-auth]: module "sql" returns ok for request 0
modcall: leaving group post-auth (returns ok) for request 0

Sending Access-Accept of id 51 to 192.168.1.50 port 2517
Session-Timeout := 1
Termination-Action := RADIUS-Request

Finished request 0

Going to the next request
--- Walking the entire request list ---
Waking up in 6 seconds...
--- Walking the entire request list ---
Cleaning up request 0 ID 51 with timestamp 46966098
Nothing to do. Sleeping until we see a request.
```

APPENDICE 3 file di configurazione di m0n0wall

```
<?xml version="1.0" ?>
<m0n0wall>
<version>1.6</version>
<lastchange>1170066343</lastchange>
<system>
<hostname>m0n0wall</hostname>
<domain>local</domain>
<dnsallowoverride />
<username>admin</username>
<password>$1$2xGLA75j$W/jiJc00HYBZX7kFjxjQv0</password>
<timezone>Etc/UTC</timezone>
<time-update-interval>300</time-update-interval>
<timeservers>193.204.114.232</timeservers>
<webgui>
<protocol>http</protocol>
<port />
</webgui>
<dnsserver>193.205.92.1</dnsserver>
<dnsserver>193.205.92.4</dnsserver>
</system>
<interfaces>
<lan>
<if>lnc1</if>
<ipaddr>192.168.69.1</ipaddr>
<subnet>24</subnet>
<media />
<mediaopt />
</lan>
<wan>
<if>lnc0</if>
<mtu />
<media />
<mediaopt />
<spoofmac />
<blockpriv />
<ipaddr>dhcp</ipaddr>
<dhcphostname />
</wan>
</interfaces>
<staticroutes />
<pppoe />
<pptp />
<bigpond />
```

APPENDICE 3 file di configurazione di m0n0wall

```
<dyndns>
<type>dyndns</type>
<username />
<password />
<host />
<mx />
<server />
<port />
</dyndns>
<dnsupdate />
<dhcpd>
<lan>
<enable />
<range>
<from>192.168.69.100</from>
<to>192.168.69.200</to>
</range>
<defaultleasetime />
<maxleasetime />
</lan>
</dhcpd>
<pptpd>
<mode />
<redir />
<localip />
<remoteip />
</pptpd>
<dnsmasq />
<snmpd>
<syslocation>Firewall di frontiera</syslocation>
<syscontact>amministratore dei rete</syscontact>
<rocommunity>public</rocommunity>
<enable />
</snmpd>
<diag>
<ipv6nat>
<ipaddr />
</ipv6nat>
</diag>
<bridge />
<syslog />
<nat />
<filter>
<rule>
<type>pass</type>
<interface>lan</interface>
```

APPENDICE 3 file di configurazione di m0n0wall

```
<source>
<network>lan</network>
</source>
<destination>
<any />
</destination>
<disabled />
<descr>Default LAN -> any</descr>
</rule>
<rule>
<type>pass</type>
<interface>lan</interface>
<protocol>tcp/udp</protocol>
<source>
<network>lan</network>
</source>
<destination>
<any />
<port>53</port>
</destination>
<descr>LAN -> DNS</descr>
</rule>
<rule>
<type>pass</type>
<interface>lan</interface>
<protocol>tcp</protocol>
<source>
<network>lan</network>
</source>
<destination>
<any />
<port>80</port>
</destination>
<descr>LAN --> HTTP</descr>
</rule>
<rule>
<type>pass</type>
<interface>lan</interface>
<protocol>tcp</protocol>
<source>
<network>lan</network>
</source>
<destination>
<any />
<port>443</port>
</destination>
```

APPENDICE 3 file di configurazione di m0n0wall

```
<descr>LAN --> HTTPS</descr>
</rule>
<rule>
<type>pass</type>
<interface>lan</interface>
<protocol>icmp</protocol>
<source>
<network>lan</network>
</source>
<destination>
<any />
</destination>
<descr>LAN --> PING</descr>
</rule>
<rule>
<type>pass</type>
<interface>lan</interface>
<protocol>tcp</protocol>
<source>
<network>lan</network>
</source>
<destination>
<any />
<port>22</port>
</destination>
<descr>LAN --> SSH</descr>
</rule>
</filter>
<ipsec />
<aliases />
<proxyarp />
<wol />
<_captiveportal>
  <page>
```

```
  <htmltext>PGh0bWw+Cjxib2R5Pgo8Zm9ybSBtZXRob2Q9InBvc3QiIGFjdGlvbj0iJFB
  PUIRBTF9BQ1RJT04kIj4KPGlucHV0IG5hbWU9ImF1dGhfdXNlciIgdHlwZT0idGV4d
  CI+CjxpbnB1dCBuYW11PSJhdXRoX3Bhc3MiIHR5cGU9InBhc3N3b3JkIj4KPGlucHV
  0IG5hbWU9InJlZGlydXJsIiB0eXBIPSJoaWRkZW4iIHZhbHVlPSIkUE9SVEFMX1JFR
  EISVVJMJCI+CjxpbnB1dCBuYW11PSJhY2NlcHQiIHR5cGU9InN1Ym1pdCIgdmFsd
  WU9IiAgT0sgICI+CjwvZm9ybT4KPC9ib2R5Pgo8L2h0bWw+Cg==</htmltext>
```

```
</page>
<timeout>60</timeout>
<interface>lan</interface>
<maxproc />
<idletimeout>5</idletimeout>
```

APPENDICE 3 file di configurazione di m0n0wall

```
<auth_method>local</auth_method>
<reauthenticateacct />
<httpsname />
<certificate />
<private-key />
<redirurl />
<radiusip />
<radiusip2 />
<radiusport />
<radiusport2 />
<radiusacctport />
<radiuskey />
<radiuskey2 />
<radiusvendor>default</radiusvendor>
<user>
<name>fausto</name>
<fullname />
<expirationdate />
<password>292f5f5f7460fc943639ec9e3b0dbe7f</password>
</user>
<bwdefaultdn />
<bwdefaultup />
<radmac_format>default</radmac_format>
</captiveportal>
<shaper>
<enable />
<pipe>
<descr>m_Total Upload</descr>
<bandwidth>144</bandwidth>
</pipe>
<pipe>
<descr>m_Total Download</descr>
<bandwidth>76</bandwidth>
</pipe>
<queue>
<descr>m_High Priority #1 Upload</descr>
<targetpipe>0</targetpipe>
<weight>50</weight>
</queue>
<queue>
<descr>m_High Priority #2 Upload</descr>
<targetpipe>0</targetpipe>
<weight>30</weight>
</queue>
<queue>
<descr>m_High Priority #3 Upload</descr>
```

APPENDICE 3 file di configurazione di m0n0wall

```
<targetpipe>0</targetpipe>
<weight>15</weight>
</queue>
<queue>
<descr>m_Bulk Upload</descr>
<targetpipe>0</targetpipe>
<weight>4</weight>
</queue>
<queue>
<descr>m_Hated Upload</descr>
<targetpipe>0</targetpipe>
<weight>1</weight>
</queue>
<queue>
<descr>m_Bulk Download</descr>
<targetpipe>1</targetpipe>
<weight>30</weight>
</queue>
<queue>
<descr>m_Hated Download</descr>
<targetpipe>1</targetpipe>
<weight>10</weight>
</queue>
<queue>
<descr>m_High Priority Download</descr>
<targetpipe>1</targetpipe>
<weight>60</weight>
</queue>
<rule>
<descr>m_TCP ACK Upload</descr>
<targetqueue>2</targetqueue>
<interface>wan</interface>
<direction>out</direction>
<source>
<any />
</source>
<destination>
<any />
</destination>
<iplen>0-80</iplen>
<protocol>tcp</protocol>
<tcpflags>ack</tcpflags>
</rule>
<rule>
<descr>m_Small Pkt Upload</descr>
<targetqueue>0</targetqueue>
```

APPENDICE 3 file di configurazione di m0n0wall

```
<interface>wan</interface>
<direction>out</direction>
<source>
<any />
</source>
<destination>
<any />
</destination>
<iplen>0-100</iplen>
</rule>
<rule>
<descr>m_Outbound DNS Query</descr>
<targetqueue>0</targetqueue>
<interface>wan</interface>
<direction>out</direction>
<source>
<any />
</source>
<destination>
<any />
<port>53</port>
</destination>
<protocol>udp</protocol>
</rule>
<rule>
<descr>m_AH Upload</descr>
<targetqueue>0</targetqueue>
<interface>wan</interface>
<direction>out</direction>
<source>
<any />
</source>
<destination>
<any />
</destination>
<protocol>ah</protocol>
</rule>
<rule>
<descr>m_ESP Upload</descr>
<targetqueue>0</targetqueue>
<interface>wan</interface>
<direction>out</direction>
<source>
<any />
</source>
<destination>
```


APPENDICE 3 file di configurazione di m0n0wall

```
<any />
</destination>
<protocol>esp</protocol>
</rule>
<rule>
<descr>m_GRE Upload</descr>
<targetqueue>0</targetqueue>
<interface>wan</interface>
<direction>out</direction>
<source>
<any />
</source>
<destination>
<any />
</destination>
<protocol>gre</protocol>
</rule>
<rule>
<descr>m_ICMP Upload</descr>
<targetqueue>1</targetqueue>
<interface>wan</interface>
<direction>out</direction>
<source>
<any />
</source>
<destination>
<any />
</destination>
<protocol>icmp</protocol>
</rule>
<rule>
<descr>m_Catch-All Upload</descr>
<targetqueue>3</targetqueue>
<interface>wan</interface>
<direction>out</direction>
<source>
<any />
</source>
<destination>
<any />
</destination>
</rule>
<rule>
<descr>m_ICMP Download</descr>
<targetqueue>7</targetqueue>
<interface>wan</interface>
```

APPENDICE 3 file di configurazione di m0n0wall

```
<direction>in</direction>
<source>
<any />
</source>
<destination>
<any />
</destination>
<protocol>icmp</protocol>
</rule>
<rule>
<descr>m_Small Pkt Download</descr>
<targetqueue>7</targetqueue>
<interface>wan</interface>
<direction>in</direction>
<source>
<any />
</source>
<destination>
<any />
</destination>
<iplen>0-100</iplen>
</rule>
<rule>
<descr>m_AH Download</descr>
<targetqueue>7</targetqueue>
<interface>wan</interface>
<direction>in</direction>
<source>
<any />
</source>
<destination>
<any />
</destination>
<protocol>ah</protocol>
</rule>
<rule>
<descr>m_ESP Download</descr>
<targetqueue>7</targetqueue>
<interface>wan</interface>
<direction>in</direction>
<source>
<any />
</source>
<destination>
<any />
</destination>
```

APPENDICE 3 file di configurazione di m0n0wall

```
<protocol>esp</protocol>
</rule>
<rule>
<descr>m_GRE Download</descr>
<targetqueue>7</targetqueue>
<interface>wan</interface>
<direction>in</direction>
<source>
<any />
</source>
<destination>
<any />
</destination>
<protocol>gre</protocol>
</rule>
<rule>
<descr>m_Catch-All Download</descr>
<targetqueue>5</targetqueue>
<interface>wan</interface>
<direction>in</direction>
<source>
<any />
</source>
<destination>
<any />
</destination>
</rule>
<magic>
<maxup>160</maxup>
<maxdown>80</maxdown>
</magic>
</shaper>
</m0n0wall>
```

Bibliografia

Bibliografia

[1] RFC 2865, C. Rigney, S. Willens, A. Rubens, W. Simpson, *Remote Authentication Dial In User Service (RADIUS)*, June 2000

<http://www.rfc-editor.org/>

[2] RFC 3588, P. Calhoun, J. Loughney, E. Guttman, G. Zorn, J. Arkko, *Diameter Base Protocol*, September 2003

<http://www.rfc-editor.org/>

[3] RFC 2903, C. de Laat, G. Gross, L. Gommans, J. Vollbrecht, D. Spence, *Generic AAA Architecture*, August 2000

<http://www.rfc-editor.org/>

[4] RFC 2904, J. Vollbrecht, P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege, D. Spence, *AAA Authorization Framework*, August 2000

<http://www.rfc-editor.org/>

[5] RFC 3460, B. Moore, Ed., *Policy Core Information Model (PCIM) Extensions*, January 2003

<http://www.rfc-editor.org/>

[6] RFC 2975 B. Aboba, J. Arkko, D. Harrington, *Introduction to Accounting Management*, October 2000

<http://www.rfc-editor.org/>

[7] RFC 1334 B. Lloyd, W. Simpson, *PPP Authentication Protocols*, October 1992

<http://www.rfc-editor.org/>

Bibliografia

[8] RFC 1994 W. Simpson *PPP Challenge Handshake Authentication Protocol (CHAP)*, August 1996

<http://www.rfc-editor.org/>

[9] RFC 3580, P. Congdon, B. Aboba, A. Smith, G. Zorn, J. Roese, *IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Usage Guidelines*, September 2003

<http://www.rfc-editor.org/>

[10] RFC 3748 B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, H. Levkowitz, Ed., *Extensible Authentication Protocol (EAP)*, June 2004

<http://www.rfc-editor.org/>

[11] RFC 2866 C. Rigney, *RADIUS Accounting*, June 2000

[12]www.freeradius.org

Altre informazioni sono state attinte da:

Jonathan Hassell, *Radius*, O'Reilly, October 2002

<http://www.javvin.com/>