

IKEV2 TESTING

DAVIDE DE ZUANE & RAHMI EL MECHRI

CONTENTS

1 Introduction 3

1.1 IPsec 3

1.2 IKE 3

1.3 Strongswan 3

2 Setup 4

2.1 Environment 4

2.2 Configuration 4

3 Testing 6

3.1 Time 6

3.2 Performance 6

3.3 Results 6

4 Conslusioni 6

A Certificati 7

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

1 INTRODUCTION

L'aumento della connettività e n

La sicurezza negli ultimi anni sta diventando sempre più importante per le comunicazioni,

c'è sempre più bisogno di realizzare collegamenti sicuri, uno strumento molto utile a questo fine sono le VPN.

1.1 IPsec

IPsec (Internet Protocol Security) è una suite di protocolli che fornisce sicurezza alle comunicazioni Internet a livello IP. L'uso attuale più comune di IPsec è quello di fornire una rete privata virtuale (VPN). IKE (Internet Key Exchange) Internet Key Exchange) è il protocollo di negoziazione e gestione delle chiavi più comunemente utilizzato per fornire chiavi negoziate e aggiornate dinamicamente per IPsec. Introduzione a quello che è ipsec e suo funzionamento

1.2 IKE

Importanza di IKE per effettuare lo scambio di chiavi per stabilire la SA.

Concetti su cui si basa IKE

1.3 Strongswan

Parlare rapidamente di quello che è quello che è anche l'architettura di strongswan charon.

2 SETUP

Andiamo a vedere nel dettaglio l'ambiente e la configurazione che abbiamo utilizzato per realizzare i test. Per verificare le capacità di IKE abbiamo previsto:

- 3 modalità di autenticazione;
- 2 chiper suite differenti da utilizzare.

Nella fase di sperimentazione abbiamo utilizzato le seguenti convenzioni:

- **Initiator:** l'host che invia la richiesta di stabilire una SA;
- **Responder:** l'host che risponde alle richieste.

2.1 Environment

Per simulare i due host della comunicazione abbiamo creato due macchine virtuali tramite l'utilizzo di qemu/kvm, questo per avere delle performance il più possibile simili a quelle reali. Le due macchine virtuali sono state create in modalità bridge, questo per evitare problemi con la modalità NAT. In questo modo le due macchine appartengono alla stessa rete e questo ci fa facilitare la configurazione. Le macchine virtuali utilizzate hanno le seguenti specifiche:

- *Processore:* 2 core
- *Memoria:* 2048MB
- *OS:* Debian 11
- *Network:* Bridge

Setup delle macchine virtuali tramite qemu cli

Dopo le macchine virtuali la lista dei vari requirements da installare

Fare un file txt contenente tutti i pacchetti da installare in modo tale da lanciare un solo apt-get install

2.2 Configuration

I file e le directory coinvolte nel processo di configurazione sono i seguenti. Dato che una delle principali modifiche di IKEv2 rispetto alla versione precedente è la possibilità di autenticazione tramite certificati andremo ad usare le directory specificate.

```

/etc
├── ipsec.conf
├── ipsec.secrets
└── ipsec.d
    ├── cacerts
    ├── certs
    └── private

```

- Il file `ipsec.conf` specifica la maggior parte delle configurazioni e le informazioni di controllo per il sottosistema IPsec (ulteriori specifiche e sintassi sono disponibili al seguente [link](#)).
- Il file `ipsec.secrets` contiene i segreti che poi verranno utilizzati nella fase di autenticazione (ulteriori specifiche al seguente [link](#)).

Certificati

Una delle principali novità che introduce IKEv2 è la possibilità di eseguire l'autenticazione tra certificati X.509. In fase di testing abbiamo preso in considerazione due tipi di certificati:

- Certificati RSA
- Certificati ECDSA

A partire da una chiave pubblica è necessario realizzare un certificato di chiave pubblica e questo richiede la chiave privata di una CA. Nel nostro caso ci siamo creati dei certificati da CA e li abbiamo distribuiti manualmente tra i due host.

Per la generazione abbiamo utilizzato il tool [pki](#)

CA Certificate

Partiamo con la generazione dei certificati da Certification Authority, di seguito sono riportati i due comandi da utilizzare. Ne occorrono due poichè per firmare i certificati ECDSA occorre una chiave con lo stesso schema.

```
$ pki --gen --type rsa --size 2048 --outform pem > 'ca.rsa.key.pem'
$ pki --gen --type ecdsa --size 256 --outform pem > 'ca.ecdsa.key.pem'
```

Ora utilizziamo la chiave privata per firmare il certificato di chiave pubblica.

```
$ pki --self --ca --lifetime 3650 --in 'ca.<type>.key.pem' --type <type> \
--dn "CN=CA" --outform pem > ca.<type>.cert.pem
```

Occorre poi distribuire questi due certificati ai due host, vanno messi all'interno della directory cacerts.

Host Certificate

Passiamo ora a generare i certificati che gli host andranno ad utilizzare nella fase di autenticazione, occorre generare la coppia chiave privata, chiave pubblica.

```
$ pki --gen --type ecdsa --size 256 --outform pem > 'host.ecdsa.key'
$ pki --gen --type rsa --size 2048 --outform pem > 'host.rsa.key'
```

E' buona norma salvare le chiavi all'interno della directory private. Ora andiamo ad estrarre la chiave pubblica da quella appena genrata e la firmiamo con la chiave delle CA del passo precedente.

```
$ pki --pub --in 'host.rsa.key' --type rsa | pki --issue --lifetime 1825 \
--cacert 'ca.rsa.cert.pem' --cakey 'ca.rsa.key.pem' \
--dn "CN=<Host_IP>" --san @<Host_IP> --san <Host_IP> \
-- flag serverAuth --outform pem > 'host.rsa.cert.pem'
```

Si procede in maniera analoga con le opportune modifiche anche per il certificato ECDSA. Questi vanno poi posizionati all'interno della directory certs.

2.2.1 Mschap

2.2.2 RSA

2.2.3 ECDSA

3 TESTING

Per misurare i cicli macchina abbiamo utilizzato perf
per installarlo apt-get install linux-perf se da problemi con workload failed è a causa dei permessi e per risolverlo basta sovrascrivere il contenuto di
/proc/sys/kernel/perf_event_paranoid per fare il report di tutto l'ambiente utilizzare perf report

3.1 Time

3.2 Performance

3.3 Results

4 CONSLUSIONI

A CERTIFICATI

Spiegare quella che è una root CA

Come funziona la catena di certificati

Spiegare quelli che sono i campi all'interno di un certificato

Cofronto tra certificato RSA e ECDSA

Per verificare le informazioni contenute all'interno di un certificato utilizzare il seguente comando:

```
openssl x509 --in <certificate>
```

REFERENCES