



UNIVERSITÀ DEGLI STUDI ROMA TRE

Dipartimento di Ingegneria
Corso di Laurea in Ingegneria Informatica

Tesi Di Laurea

Realizzazione di un sistema di
cyber-defense: utilizzo delle VPN per un
accesso remoto sicuro a risorse interne

Laureando

Filippo Visconti

Matricola 547344

Relatore

Prof. Maurizio Patrignani

Correlatore

Federico Lommi

Anno Accademico 2021/2022

Questa è la dedica

Ringraziamenti

Questi sono i ringraziamenti.

Introduzione

Questa è l'introduzione.

Pericoli di esporre un server su internet

The Security Architecture of the OSI Reference Model (ISO 7498-2) considers five main classes of security services: authentication, access control, confidentiality, integrity and non-repudiation. These services are defined as follows: The authentication service verifies the supposed identity of a user or a system. The access control service protects the system resources against non-authorized users. The confidentiality service protects the data against non-authorized revelations. The integrity service protects the data against non-authorized modifications, insertions or deletions. The non-repudiation service prevents an entity from denying previous commitments or actions.

Termini di base

Computer networks there are a variety of the following types of computer network based on scope. The scope here is how big the computer network will be built. Based on spaces in scope, a computer network can be distinguished into two, namely [1] : a) Local Area Network (LAN), is a computer network which is built in the room a small scope as a single building or group of buildings. LAN is built in a limited scope and usually owned by organizations that already have the devices installed. An internal data rate of the LAN is usually much greater than the WAN. Wide Area Network (WAN), is a network that covers a large geographic area requires delimiters and rely at least partly on the circuit provided by public operators. Typically, a WAN consists of a number

of switching node interconnects. A transmission from one of the devices is channeled through the internal node to the device purpose. This node (including node limit) does not affect the contents of the data, their goal was to provide a switching facility will move data from node to node until they reach their destination. Traditionally, WAN has been implemented using one of the two technologies: circuit switching and packet switching. Recently, frame relay and ATM networks have assumed the lead role which uses it [2].

2.2 Virtual Private Network Virtual Private Network (VPN) is a computer network where connections between its nodes utilize public networks (internet/WAN) as it may be in certain cases or conditions do not allow it to build its own infrastructure. When the Connect VPN, the interconnection between the node such as an independent network that has actually created a special line pass through connection or a public network. At every company site, workstations, servers, and databases connected by one or more local area network (LAN) a LAN is under the control of the network manager and can be configured and tuned for cost-effective. The Internet or other public networks can be used to connect the sites, provide cost savings over the use of private networks and reduction of the burden of wide area network traffic to providers of public networks [2].

PILA ISO OSI

TCP is the main protocol in TCP/IP networks. The IP protocol process data packets while TCP allow two hosts to exchange data streams and establish a connection. TCP guarantees that packets will arrive their destination in the same order in which they were sent [7]. UDP provides unreliable, minimum, best-effort, message delivery to upper-layer protocols and applications. UDP do not setup a permanent connection between two end points [8].

The adjustments between TCP and UDP regardless of VPN usage is always said to be the same: Speed is sacrifice for reliability as UDP is connectionless and the server sending the data theoretically does not ensure if it reaches the destination or not. TCP is a connection-oriented protocol, which implies that end-to-end communications is set up using handshaking. Once the connection is established, data can be transferred bi-directionally over the link. UDP is a connectionless protocol and therefore less complex message based when compared to TCP, which includes that the point-to-point

connection is not dedicated and data is transferred uni-directional from the source to its destination without checking whether the receiver is active. TCP regulate retransmission, message acknowledgment, and timeout. TCP deliver lost messages along the way upon multiple attempts. In TCP, there is no missing data, and if ever there are multiple timeouts, the connection is dropped. When a UDP message is sent there is no guarantee that the message will reach its destination; it could get dropped along the way. There is no retransmission, timeout and acknowledgment. When two data packets are sent in sequence, the first message will reach the destination first. When data segments arrive in the wrong order, TCP buffers hold the data until all data are re-ordered before being transmitted; when using UDP the order in which messages arrive cannot be predicted. When TCP packets are transmitted from one end to a remote end across the network, the data packets are reordered in the same sequence created by the sender. The protocol notifies when segments of the data stream have been corrupted, reordered, discarded or duplicated by the network. TCP is a reliable protocol as the sender can retransmit damaged segments. However retransmission creates latency.

Necessità di un'infrastruttura di rete sicura

Ancora del testo. Come si afferma i

Organizzazione dei capitoli

Ancora del testo. Come si afferma i

Indice

Introduzione	iv
Pericoli di esporre un server su internet	iv
Termini di base	iv
Necessità di un'infrastruttura di rete sicura	vi
Organizzazione dei capitoli	vi
Indice	vii
Elenco delle figure	xi
1 Requisiti	1
1.1 Caratteristiche della rete aziendale	1
1.1.1 Diagramma di rete	1
1.1.2 Descrizione dei componenti fondamentali	2
1.1.3 Servizi offerti all'esterno	3
1.1.4 Servizi offerti all'interno	4
1.2 Necessità degli utenti	6
1.2.1 Accesso ai servizi interni senza esposizione all'esterno	6
1.3 Requisiti di sicurezza	6
1.3.1 Controllo del traffico	6
1.3.2 Trasmissione sicura dei dati	6
1.3.3 Controllo dei dispositivi	6
1.3.4 Compatibilità	7

2	Stato dell'arte	8
2.1	Virtual Private Networks	8
2.1.1	Architetture disponibili	8
2.1.2	Perché soddisfano i requisiti	9
2.1.3	Soluzioni principali	9
2.2	Internet Protocol Security	10
2.2.1	Panoramica	10
2.2.2	Transport mode vs Tunnel mode	10
2.2.3	Protocolli utilizzati	11
2.2.4	Cifratura	13
2.2.5	Autenticazione	14
2.2.6	Implementazioni	14
2.2.7	Considerazioni	14
2.3	PPTP	14
2.3.1	Panoramica	14
2.3.2	Protocolli utilizzati	15
2.3.3	Cifratura	15
2.3.4	Autenticazione	15
2.3.5	Considerazioni	16
2.4	OpenVPN	16
2.4.1	Panoramica	16
2.4.2	Protocolli utilizzati	17
2.4.3	Tunnel TCP vs UDP	18
2.4.4	Cifratura	18
2.4.5	Autenticazione	19
2.4.6	Misure di sicurezza aggiuntive	19
2.4.7	Considerazioni	20
2.5	WireGuard	20
2.5.1	Panoramica	20
2.5.2	Protocolli utilizzati	21
2.5.3	Cifratura	21

2.5.4	Autenticazione	21
2.5.5	Considerazioni	22
3	Realizzazione	23
3.1	Virtualizzatore	23
3.1.1	Caratteristiche e funzionamento	23
3.1.2	VirtualBox	24
3.1.3	VMWare ESXi	24
3.1.4	Installazione e configurazione dei servizi VPN	24
3.2	Installazione e configurazione di IPSec - tunnelmode	25
3.2.1	Aggiunta dell'interfaccia di rete virtuale	25
3.2.2	Installazione di strongSwan	25
3.2.3	Configurazione di strongSwan	26
3.2.4	Creazione del certificato per un client	26
3.3	Installazione e configurazione di OpenVPN over TCP	27
3.3.1	Installazione di openvpn	27
3.3.2	Certificati	28
3.3.3	Configurazione del profilo VPN per un client	28
3.4	Installazione e configurazione WireGuard	28
3.4.1	Configurazione del profilo VPN per un client	28
3.5	Configurazione del Firewall	29
4	Testing	30
4.1	Modalità di esecuzione dei test	30
4.1.1	Panoramica di iPerf3	30
4.1.2	Panoramica di mrt	30
4.1.3	Scelta della configurazione di test	32
4.1.4	Criteri di valutazione	32
4.2	Misure senza VPN	35
4.3	Misure con IPSec e IKEv2	35
4.4	Misure con OpenVPN over TCP	35
4.5	Misure con WireGuard	35

4.6	Analisi delle misure	35
5	Security concerns	38
5.1	Principali problematiche di sicurezza	38
5.2	Attacchi mirati agli utenti	38
5.3	Attacchi mirati al sistema	38
5.4	Multi-factor authentication	38
	Conclusioni e sviluppi futuri	40
	Quale è uscito vincitore	40
	Come migliorare le misure	40
	Test di VPN Peer-To-Peer	40
	Zero Tier	40
	Bibliografia	41

Elenco delle figure

1.1	Diagramma di rete	1
2.1	Modello logico di connessione con e senza VPN	9
2.2	Transport mode vs Tunnel mode diagram	10
2.3	Authentication Header packet formats	11
2.4	Encapsulating Security Payload packet formats	12
2.5	Dettaglio di un pacchetto OpenVPN	17
3.1	Installazione fisica vs Installazione virtuale	24
3.2	Dettagli interfaccia virtuale strongswan0	25
3.3	Architettura di installazione di WireGuard	29
4.1	Esempio di output di iPerf3	31
4.2	Esempio di output di mrt	31
4.3	Throughput senza VPN su 300 secondi	35
4.4	IPSec Throughput su 300 secondi	35
4.5	OpenVPN Throughput su 300 secondi	36
4.6	WireGuard Throughput su 300 secondi	36
4.7	Confronto dei throughput grezzi	36
4.8	Confronto dei throughput raffinati	37
4.9	Confronto dei throughput medi	37

Capitolo 1

Requisiti

1.1 Caratteristiche della rete aziendale

La rete su cui siamo stati chiamati a lavorare è illustrata nel diagramma seguente.

1.1.1 Diagramma di rete

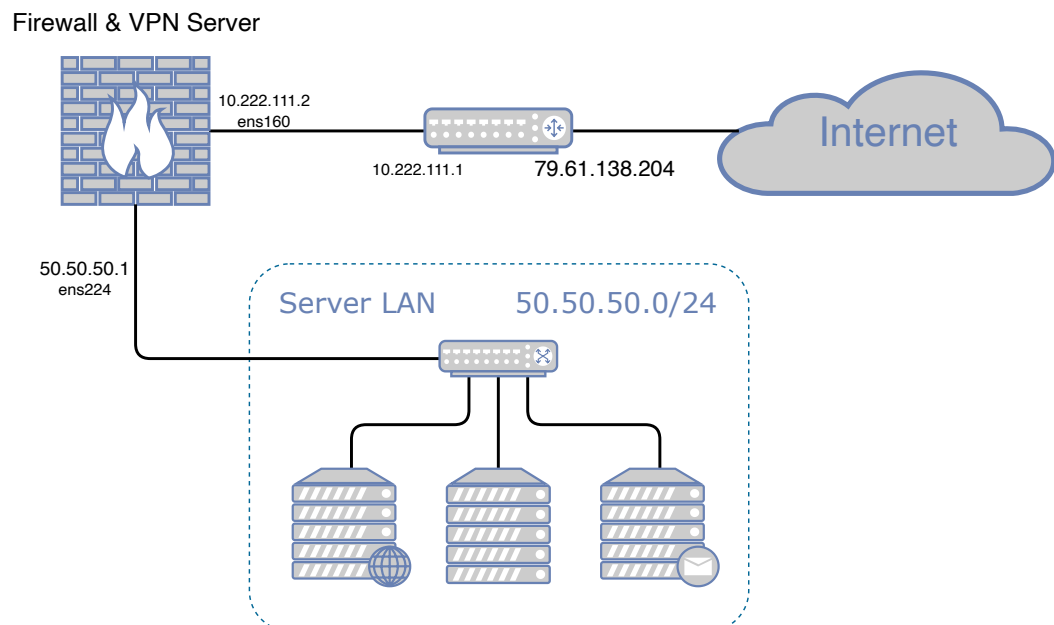


Figura 1.1: Diagramma di rete

La porzione di diagramma che rappresenta l'infrastruttura di rete dell'azienda è quella sinistra della nuvola. La porzione destra raffigura, invece, un'altra sottorete, che per fini di testing immaginiamo sia la rete casalinga di un dipendente dell'azienda. La nuvola sta a rappresentare tutta la rete Internet.

1.1.2 Descrizione dei componenti fondamentali

1.1.2.1 Router

Un componente essenziale all'interno dell'infrastruttura di rete è il router. Il router è un dispositivo di rete che lavora al livello 3 del modello OSI e che permette e gestisce l'instradamento dei pacchetti tra sottoreti diverse. Le tabelle d'instradamento, salvate nella memoria interna del router, contengono informazioni che riguardano come raggiungere gli altri nodi della rete e sono lo strumento che permette al router di instradare correttamente i pacchetti. Queste tabelle associano il prefisso IP [Pos81, RFC0791] e relativa maschera della sottorete di destinazione con il *next-hop*, l'indirizzo IP del prossimo router a cui deve essere destinato al pacchetto affinché si avvicini alla sua vera destinazione, e l'interfaccia di rete del router da cui il pacchetto deve essere inoltrato affinché possa raggiungere il *next-hop*. Generalmente la funzione di routing è svolta da un componente hardware dedicato, che, se di fascia alta, permette di raggiungere prestazioni pari alla velocità della linea - ossia, spedisce i pacchetti alla stessa velocità alla quale li riceve. Tuttavia, è possibile il compito venga svolto da server generici, a patto che siano dotati di un numero adatto di schede di rete, su cui gira un software apposito.

1.1.2.2 Firewall

In entrambi i casi, è comune che il router abbia un firewall [Fre00] integrato. Un firewall è un dispositivo fisico, o un software, che ha come obiettivo la regolazione del traffico di una rete. Ciò avviene applicando una serie di regole che coinvolgono lo stato, la porta e il protocollo dei pacchetti che lo attraversano. L'amministratore di rete ha la responsabilità di inserire regole appropriate al contesto, affinché, tutto ciò che non è strettamente necessario, non venga fatto passare. Un esempio di Firewall software molto conosciuto in ambienti UNIX, è `iptables`. Il seguente comando mostra una

regola che *consente* il passaggio di un pacchetto in entrata sul firewall dalla scheda di rete `eth0`, destinato alla porta 22 TCP, e che sia il primo di una comunicazione, o faccia parte di una comunicazione già instaurata.

```
iptables -A INPUT -p tcp --dport 22 -i eth0 \  
-m state --state NEW,ESTABLISHED -j ACCEPT
```

Il sistema a disposizione avrà un router/firewall installato su un server, che ha come sistema operativo CentOS 7 [Cen] - la versione gratuita di Red Hat Enterprise Linux [Lin].

1.1.2.3 Le LAN utilizzate

Nella configurazione della rete aziendale corrente è presente soltanto una sottorete, denominata LAN dei server, dove risiedono esclusivamente i server che erogano servizi all'esterno dell'azienda. Tutti i servizi per uso interno, destinati a una ulteriore LAN, in questo esempio sono erogati dal router/firewall stesso.

1.1.3 Servizi offerti all'esterno

L'azienda ha necessità di pubblicare

- un sito web, il cui hosting è effettuato sul web server interno;
- un mail server, che si occupa di inviare e ricevere i messaggi di posta elettronica

1.1.3.1 Web servers

Il sito web è servito in HTTP sulla porta 80 e in HTTPS sulla porta 443, e la pubblicazione è affidata a un noto servizio, *Apache Httpd* [Apa].

1.1.3.2 Mail servers

Ancora del testo. Come si afferma in [JS96] molto lavoro deve ancora essere fatto.

1.1.4 Servizi offerti all'interno

1.1.4.1 DHCP server

Il Dynamic Host Configuration Protocol [Dro97, RFC2131] è un protocollo ausiliario che permette l'assegnazione automatica degli indirizzi IP e altri parametri di configurazione ai dispositivi connessi alla rete usando una architettura client-server. Il DHCP offre un servizio non connesso e utilizza UDP come protocollo di trasporto. Il server ascolta le richieste (che saranno broadcast, destinate per convenzione all'IP 255.255.255.255) sulla porta 67 UDP, e inoltra le risposte al client sulla porta 68 UDP. Altri parametri di configurazione che comunemente accompagnano l'appena assegnato indirizzo IP sono i server DNS [Moc87, RFC1034] di default, l'indirizzo IP del default gateway, e la durata per il quale l'IP assegnato è valido.

1.1.4.2 DNS server

Il Domain Name System è il sistema di assegnazione gerarchico e decentralizzato dei nomi che identificano gli host in rete. Un'analogia che aiuta a comprendere la funzione del DNS è quella della rubrica telefonica. Infatti, come nella rubrica telefonica viene mantenuta un'associazione tra un nome - facile da ricordare per una persona - e il relativo numero di telefono - più difficile da ricordare, e facile da confondere -, così il DNS conserva dei *resource records* composti da un nome - **example.com** - associato a un indirizzo IPv4 o IPv6 - **93.130.23.53**. Si tratta di un protocollo di livello 7, che generalmente comunica sulla porta 53 UDP, ma potrebbe sfruttare anche VPN o tunnel, TLS, HTTPS, Tor. È una potenzialità interessante, in quanto le richieste non sono crittate e si potrebbe andare incontro a problemi di sicurezza. Il DNS è in grado di memorizzare anche altre informazioni riguardanti un certo dominio, tra cui:

- i *name servers* che sono autorità per quel dominio - coloro che a loro volta memorizzano i resource records dei vari sottodomini;
- gli indirizzi IP dei mail exchanger di riferimento per quel dominio;
- degli alias, ossia un'associazione tra due nomi di dominio.

Nella configurazione corrente, il server DNS, che gira sullo stesso server del Firewall, lavora come relay e il suo IP viene distribuito a tutti i client della rete interna via DHCP come DNS resolver. Ciò significa che tutti gli host della rete, nel momento in cui devono risolvere un nome, inviano una richiesta al server DNS interno, che si occuperà lui di risolverlo e, una volta ottenuto il risultato, lo restituisce al richiedente. Questo comporta diversi vantaggi, tra cui:

- la comunicazione verso l'esterno per la risoluzione dei nomi avviene da un unico punto della rete;
- si può fare caching, ossia mantenere in memoria per un certo periodo di tempo (che viene specificato nella risposta che il server DNS riceve) le risposte delle varie risoluzioni, cosicché, se di una richiesta si era già trovata la risposta, non dovrà essere fatta di nuovo la risoluzione;
- si possono implementare dei filtri per bloccare la risoluzione di nomi a cui si vuole limitare l'accesso;
- si possono facilmente loggare le varie richieste.

1.1.4.3 Web app interne

All'interno della rete locale dell'azienda, sono accessibili degli applicativi che permettono la gestione di alcuni sistemi, ad esempio degli apparati di rete.

1.1.4.4 File servers

Affinché i dipendenti autorizzati possano collaborare e accedere a file condivisi, è stato predisposto un file server, a cui si può accedere con protocolli quali FTP [PR85, RFC0791] e SFTP. Tuttavia, i file in questione possono contenere dati sensibili. Per questo motivo, è necessario che la risorsa sia adeguatamente protetta, non esposta alla rete esterna e che l'accesso sia regolamentato.

1.1.4.5 Database servers

Similmente al file server, c'è anche un database server a disposizione dei dipendenti, le cui necessità di sicurezza rispecchiano quelle del file server.

1.2 Necessità degli utenti

1.2.1 Accesso ai servizi interni senza esposizione all'esterno

Router, Firewall, IDS, IPS, VPN. Ancora del testo. Come si afferma in [JS96] molto lavoro deve ancora essere fatto.

1.2.1.1 Remote work

Ancora del testo. Come si afferma in [JS96] molto lavoro deve ancora essere fatto.

1.3 Requisiti di sicurezza

1.3.1 Controllo del traffico

1.3.1.1 Proxy interno obbligatorio

Ancora del testo. Come si afferma in [JS96] molto lavoro deve ancora essere fatto.

1.3.2 Trasmissione sicura dei dati

Ancora del testo. Come si afferma in [JS96] molto lavoro deve ancora essere fatto.

1.3.2.1 Evitare intercettazioni

Vedi Cina con il Great Firewall

1.3.3 Controllo dei dispositivi

Ancora del testo. Come si afferma in [JS96] molto lavoro deve ancora essere fatto.

1.3.3.1 Logs

Ancora del testo. Come si afferma in [JS96] molto lavoro deve ancora essere fatto.

1.3.4 Compatibilità

Deve essere compatibile con i 3 OS desktop e i 2 mobile principali

Capitolo 2

Stato dell'arte

2.1 Virtual Private Networks

Una rete privata virtuale consiste in una rete il cui accesso è regolamentato, che si appoggia a un protocollo di trasporto pubblico e condiviso, e che consente di garantire confidenzialità della comunicazione, accesso solo previa autenticazione, integrità dei dati e protezione da alcuni tipi di attacchi, ad esempio Man-in-the-middle o attacco replay.

2.1.1 Architetture disponibili

Una rete VPN può realizzare diversi tipi di collegamenti, per soddisfare esigenze diverse. Nei paragrafi successivi, si andranno ad analizzare i 3 tipi di architetture più comuni:

- Gateway-to-Gateway
- Host-to-Host
- Host-to-Gateway

2.1.1.1 Gateway-to-Gateway

Consiste in una VPN che connette in maniera stabile due reti. Questa configurazione permette ad esempio di estendere una rete privata tra diverse location geograficamente separati e distanti a piacere, oppure di garantire a una serie di uffici un accesso sicuro a un data center.

2.1.1.2 Host-to-Host

Questa configurazione è la meno comune. Consiste nello stabilire una comunicazione diretta tra due host, in cui uno fa da server VPN e l'altro da client VPN. Un caso d'uso potrebbe essere un amministratore di sistema che deve fare gestione remota di un apparecchio.

2.1.1.3 Host-to-Gateway

In questa modalità, il risultato che si ottiene è lo stesso che si avrebbe connettendo un host alla rete locale in cui risiede il server VPN. È usata principalmente per offrire un accesso sicuro da remoto alla rete. Quando l'host vuole instaurare una connessione VPN con il server, gli viene richiesto di autenticarsi.

2.1.2 Perché soddisfano i requisiti

Una VPN in configurazione Host-to-Gateway si prospetta come la soluzione più pratica e funzionale per soddisfare le necessità dell'azienda e dei suoi dipendenti, garantendo loro la possibilità di accedere alle risorse interne attraverso un canale di comunicazione privato, ad accesso controllato, criptato e dove è assicurata l'integrità dei dati.

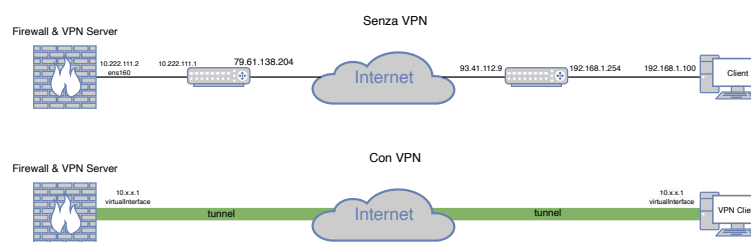


Figura 2.1: Modello logico di connessione con e senza VPN

2.1.3 Soluzioni principali

Tra le soluzioni VPN più comuni troviamo:

- Point-to-Point Tunneling Protocol
- Internet Protocol Security

- OpenVPN
- Wireguard

2.2 Internet Protocol Security

2.2.1 Panoramica

IP Security è una suite di protocolli il cui obiettivo è rendere sicura la comunicazione tra due computer attraverso una rete IP. Contiene protocolli per la mutua autenticazione degli host e per la negoziazione delle chiavi di cifratura da usare durante la sessione. In molti contesti, rendere sicuro il livello di rete (L3 OSI) è una soluzione migliore rispetto a rendere sicuro il livello di trasporto (L4 OSI) o di presentazione (L7 OSI), in quanto offre un ulteriore punto di controllo per gli amministratori e più flessibilità nell'analizzare, e gestire, ogni singolo pacchetto IP. IPsec supporta l'autenticazione a livello di rete, autenticazione del mittente, integrità dei dati, cifratura, e protezione dagli attacchi replay, protezione dall'analisi del traffico e controllo degli accessi.

2.2.2 Transport mode vs Tunnel mode

The IPsec protocols AH and ESP can be implemented in a host-to-host transport mode, as well as in a network tunneling mode.

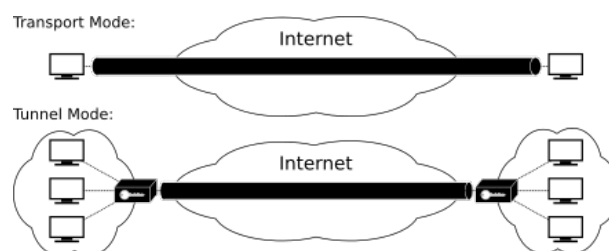


Figura 2.2: Transport mode vs Tunnel mode diagram

2.2.2.1 Transport mode

In transport mode, generalmente solo il payload del pacchetto IP è cifrato o autenticato. L'indirizzamento non cambia, dato che l'header IP non è né modificato né cifrato;

tuttavia, quanto si usa il protocollo Authentication Header - approfondito in seguito - l'indirizzo IP non può essere modificato da Network Address Translation, in quanto una modifica al campo invaliderebbe l'hash. Il livello di trasporto e di applicazione sono sempre certificati da un hash, quindi il loro contenuto non può essere modificato in alcun modo, ad esempio utilizzando una traduzione dei numeri delle porte. Un superamento delle problematiche causate dall'attraversamento di NAT è definito dalle RFC che descrivono il meccanismo NAT-T, ma che va oltre gli scopi di questa tesi.

2.2.2.2 Tunnel mode

In tunnel mode, l'intero pacchetto è cifrato e autenticato. È dunque incapsulato all'interno di un nuovo pacchetto IP con un nuovo header IP. Generando un nuovo header IP, non si incontra nessuna difficoltà nell'attraversamento di NAT.

2.2.3 Protocolli utilizzati

IPSec utilizza i seguenti protocolli per stabilire una connessione sicura. Sia AH che ESP, descritti in seguito, possono lavorare in tunnel mode o in transport mode.

2.2.3.1 Authentication Header

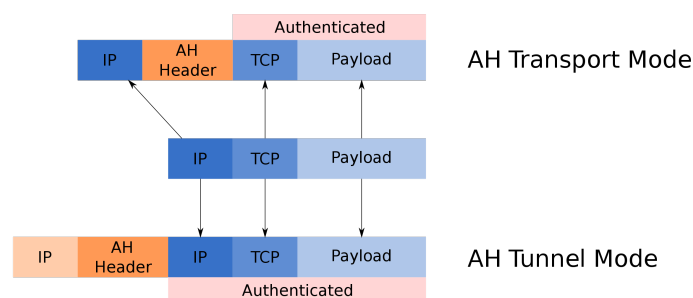


Figura 2.3: Authentication Header packet formats

Authentication Header [Ken05a, RFC4302] garantisce integrità per tutti gli header dei pacchetti, ad eccezione di alcuni campi dell'header IP, e autenticazione del mittente. Se configurato, è anche possibile utilizzarlo per offrire protezione dagli attacchi replay.

AH si interfaccia direttamente con IP, utilizzamndo il protocollo IP numero 51. AH autentica l'intero datagramma, ad eccezione dei campi variabili. Tuttavia, le informazioni contenute nel datagramma sono trasferite in chiaro e, dunque, leggibili da uno sniffer. Per questo motivo, AH non soddisfa i requisiti di sicurezza richiesti.

2.2.3.2 Encapsulating Security Payload

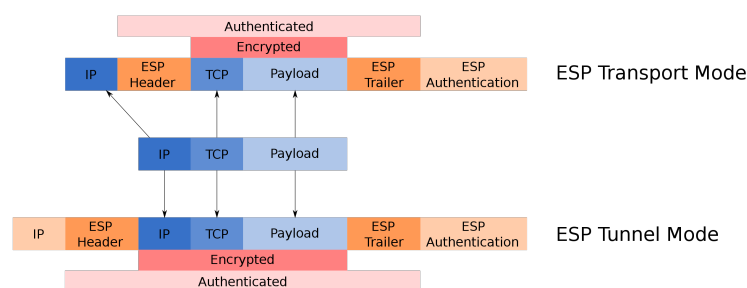


Figura 2.4: Encapsulating Security Payload packet formats

Encapsulating Security Payload [Ken05b, RFC4303] offre confidenzialità dei dati, autenticazione del mittente, controllo di integrità e protezione da attacchi relay. In Transport mode, non autentica né cifra l'header IP: cioè potrebbe esporre le informazioni contenute a potenziali attacchi mentre il pacchetto è in transito. Tuttavia, la Transport mode necessita di meno potenza computazionale, ottenendo un overhead minore della tunnel mode, rinunciando a una maggior sicurezza.

In Tunnel mode, viene creato un nuovo header IP e usato come header esterno del pacchetto, eguito dall'header ESP e poi il pacchetto originale (sia header IP che payload originale). L'ESP Trailer e gli opzionali dati di autenticazione sono aggiunti dopo il payload. Quando si usano cifratura e autenticazione contemporaneamente, ESP protegge completamente il pacchetto originale, perché diventa il payload del nuovo pacchetto ESP. Da notare è che non viene protetto il nuovo header IP. Un gateway deve necessariamente usare ESP in Tunnel mode.

2.2.3.3 Internet Key Exchange v2

Ancora del testo—IKE è un acronimo per Internet key exchange ed è il protocollo usato per stabilire una security association nella suite di protocolli IPsec. Questo protocollo è definito in RFC 4306. —

Internet Key Exchange [Kau05, RFC4306] è un protocollo che svolge la funzione di negoziazione, gestione e creazione delle Security Associations. Una SA è un insieme di regole necessarie a definire le funzionalità e i sistemi di sicurezza per stabilire una connessione IPSec. Può essere definita manualmente, anche se non scala dovutamente con VPN di grandi dimensioni. Un metodo più comune è quello di usare una delle cinque possibili modalità di scambio: main, aggressive, quick, informational e group. Le modalità sono differenti per velocità e l'uso di funzioni di cifratura. IKEv2 è la versione più recente di IKE e migliora il protocollo rendendolo più semplice, garantendo affidabilità nel recapito dei messaggi, protezione contro attacchi di tipo DenialOfService e migliora l'uso di IKE su gateways NAT. È un protocollo di livello applicazione e utilizza il protocollo UDP come protocollo di trasporto; la porta su cui viene stabilita la connessione è 500.

2.2.4 Cifratura

IPSec supporta diversi protocolli di cifratura, tra cui AES, Blowfish, Triple DES, ChaCha e DES-CBC. Inoltre, usa due tipi di cifratura: simmetrica e asimmetrica. In una codifica simmetrica, una chiave è condivisa tra gli utenti, mentre una asimmetrica fa affidamento su entrambe le chiavi pubbliche e private. La codifica asimmetrica è considerata più sicura: molti utenti condividono la chiave pubblica, ma la sicurezza fa affidamento sulla chiave privata - protetta a tutti i costi - che non ha bisogno di essere condivisa con nessuno (a differenza di una chiave simmetrica). IPSec usa la cifratura asimmetrica per instaurare una connessione sicura, per poi sfruttare quella simmetrica per migliorare la velocità di collegamento. Per quello che riguarda il collegamento, è compatibile sia con UDP che con TCP.

2.2.5 Autenticazione

L'autenticazione a chiave pubblica e privata assicura che mittenti e destinatari stiano effettivamente comunicando con il giusto partner. IPsec supporta molteplici sistemi di autenticazione, tra cui: HMAC-SHA1/SHA2, certificate authorities (CAs), RSA, ECDSA, e pre-shared key (PSK). Ogni tipologia ha i suoi pregi e difetti e casi d'uso in cui è preferibile. Ogni protocollo punta a garantire che i dati rimangano sicuri e affidabili attraverso il loro tragitto.

2.2.6 Implementazioni

StrongSwan è una implementazione open-source di IPsec per Linux. Supporta funzionalità come IPv6, certificati X.509 a chiave pubblica, liste di certificati revocati, storage di chiavi RSA private su smartcard e implementazione completa del protocollo IKEv2.re.

2.2.7 Considerazioni

Questa suite di protocolli consente di implementare una soluzione VPN accademicamente perfetta, robusta dal punto di vista della sicurezza ed efficace. L'unico impedimento che ha è che richiede l'utilizzo di due porte dedicate i due protocolli ausiliari utilizzati (AH/ESP e IKE), che potrebbero rendere l'utilizzo più difficoltoso in ambienti con firewall molto limitanti.

2.3 PPTP

2.3.1 Panoramica

Si tratta di uno dei più vecchi protocolli VPN in uso ancora oggi, ma in quanto tale ha alcune gravi criticità date dall'età. Ad esempio, la crittografia a 128 bit e il protocollo usato per l'autenticazione (MS-CHAP) contenente note vulnerabilità lo rendono ormai un protocollo insicuro, da evitare se le informazioni che transitano sono sensibili. Tuttavia, è estremamente semplice da configurare e il più veloce dal punto di vista prestazionale, il che lo rende ideale per usi quali streaming video o l'utilizzo di VPN su terminali con potenze di calcolo estremamente limitate. È stato sviluppato da Microsoft

nel 1999 [HPV⁺99, RFC2637] e lavora instaurando un canale di controllo tra i due peers sulla porta 1723 TCP e un tunnel GRE su cui transitano effettivamente i dati.

2.3.2 Protocolli utilizzati

2.3.2.1 Generic Routing Encapsulation

GRE è un protocollo di tunneling sviluppato da Cisco Systems che può incapsulare un'ampia varietà di protocolli di livello di rete all'interno di collegamenti Point-to-Point o Point-to-Multipoint virtuali su una rete IP.

2.3.3 Cifratura

Microsoft Point-to-Point Encryption (MPPE) may be used with PPTP to provide an encrypted connection but PPTP itself doesn't use encryption. MPPE uses the RC4 algorithm with either 40 or 128-bit keys. All keys are derived from the cleartext authentication password of the user. RC4 is stream cipher; therefore, the sizes of the encrypted and decrypted frames are the same size as the original frame.

Con PPTP, è possibile usare Microsoft Point-to-Point Encryption (MPPE) per instaurare una connessione cifrata, ma PPTP di base non usa cifratura. MPPE usa l'algoritmo RC4 con chiavi da 40 o 128-bit. Tutte le chiavi sono derivate dalla password in chiaro dell'utente. Tuttavia, la RFC7465 proibisce l'uso di RC4 in quanto non robusto a sufficienza

2.3.4 Autenticazione

Per quel che riguarda l'autenticazione degli utenti, PPTP può usare uno dei seguenti protocolli:

- Extensible Authentication Protocol (EAP),
- Microsoft Challenge Handshake Authentication Protocol (MSCHAP) version 1 and version 2,
- Challenge Handshake Authentication Protocol (CHAP),
- Shiva Password Authentication Protocol (SPAP),

- Password Authentication Protocol (PAP).

MSCHAP version 2 e EAP-Transport Layer Security (TLS) sono protocolli migliori rispetto agli altri supportati perché offrono mutua autenticazione, dove sia il client che il server verificano l'identità dell'altro. Se un client si autentica attraverso uno degli altri protocolli, il server verifica l'identità del client, ma il client non ha modo di verificare quella del server.

2.3.5 Considerazioni

Non offrendo una cifratura adeguata, PPTP non è una soluzione ritenuta accettabile per il caso d'uso in questione.

2.4 OpenVPN

2.4.1 Panoramica

OpenVPN è una VPN SSL che permette di incanalare tutto il traffico di una sottorete attraverso una unica porta UDP o TCP, e fa affidamento su OpenSSL. Come le altre soluzioni VPN, OpenVPN servizi essenziali di sicurezza quali autenticazione, cifratura, integrità dei dati e controllo degli accessi. Supporta due modalità di lavoro, routing e bridging:

Routing consiste nell'interconnessione di due sottoreti indipendenti, dove il server VPN (generalmente installato sul router) inoltra i pacchetti all'indirizzo IP specificato in fase di configurazione. Si tratta quindi di un collegamento a livello 3 del modello OSI.

Bridging è una modalità che lavora esclusivamente all'interno di una sottorete; il funzionamento è analogo a quello di uno switch ethernet fisico.

OpenVPN è una soluzione che lavora in user space, dunque l'overhead generato è maggiore in quanto sono necessarie molteplici copie dei pacchetti affinché siano trasferiti dal kernel space allo user space. Supporta l'intero insieme delle funzionalità di TLS, necessitando di una ampia code base, mostrando un maggior potenziale a soffrire di vulnerabilità.

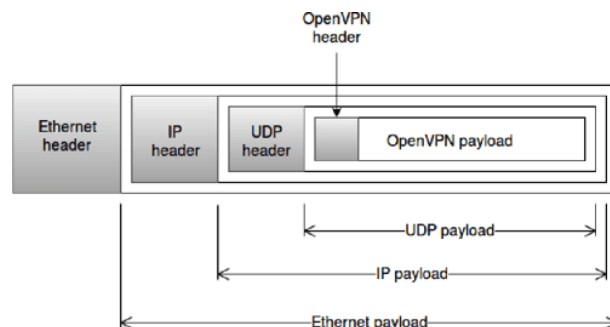


Figura 2.5: Dettaglio di un pacchetto OpenVPN

2.4.2 Protocolli utilizzati

Come precedentemente accennato, OpenVPN usa la libreria di OpenSSL, che implementa il protocollo Transport Layer Security, progettato per offrire una connessione sicura attraverso una rete non sicura. A differenza del puro TLS, OpenVPN offre all'utente la possibilità di utilizzare una pre-shared key per generare quel che è noto come HMAC firewall, che autentica tutta la sequenza di handshake TLS.

Essendo UDP un protocollo non connesso, i pacchetti IP criptati e firmati che sono incanalati tramite UDP, non hanno nessuna garanzia di affidabilità. L'affidabilità necessaria per una sicura autenticazione è garantita, però, dal protocollo TLS che utilizza TCP come protocollo di trasporto. È importante notare che il canale dati e il canale di controllo transitano all'interno dello stesso tunnel UDP (o TCP). L'incapsulamento dei pacchetti è descritto dal seguente diagramma.

La struttura mostrata si applica a tutti i pacchetti OpenVPN; tuttavia, differenti pacchetti avranno differenti payloads.

2.4.2.1 Secure Socket Layer/Transport Layer Security VPNs

Il protocollo Transport Layer Security, originariamente noto come Secure Socket Layer, è un protocollo progettato per garantire una connessione sicura attraverso una rete non sicura. TLS permette autenticazione di client e server, integrità dei dati e confidenzialità. Per l'autenticazione usa i certificati X.509 [CSF⁺08, RFC5280] con una crittografia asimmetrica e si occupa di negoziare una chiave di sessione simmetrica. Un vantaggio delle VPN SSL rispetto a quelle basate su IPsec è che riescono a lavorare anche in reti

protette da firewall molto stringenti, in quanto la maggior parte delle aziende non filtra il traffico TCP sulla porta 443, essendo normalmente usato dai dipendenti per accedere a Internet. OpenVPN di default utilizza la porta 1194 UDP, ma, nel caso quella porta fosse chiusa, può utilizzare la 443 TCP.

2.4.3 Tunnel TCP vs UDP

Premesso che attraverso i tunnel VPN passa traffico sia TCP che UDP, anche i tunnel stessi possono essere realizzati con connessioni TCP o UDP.

Il protocollo TCP utilizza notevoli algoritmi per assicurare un recapito corretto dei dati al destinatario. Avere due connessioni TCP una dentro l'altra forzerà gli algoritmi di entrambe le connessioni a lavorare in parallelo. Non essendo TCP progettato per lavorare in quella condizione, si potrebbe andare incontro a problemi quali il *retransmission problem*, *TCP meltdown* e doppia ritrasmissione. Questi problemi potrebbero verificarsi nel momento in cui entrambe le connessioni stanno tentando di ritrasmettere pacchetti.

Tutto ciò non vale per il protocollo UDP, che come descritto precedentemente, è un protocollo non connesso senza nessuna garanzia che il messaggio raggiunga correttamente il destinatario. A discapito dell'affidabilità, si possono ottenere velocità di trasmissione notevolmente superiori.

TCP potrebbe rivelarsi la scelta migliore solo nel caso in cui si debba creare un tunnel che passi attraverso una rete instabile, o attraverso una rete che applica forti censure.

2.4.4 Cifratura

OpenVPN uses a custom security protocol and SSL/TLS for key exchange. OpenSSL is used for encryption, which means a wide range of various cryptographic algorithms can be used. OVPN uses AES-based algorithms, with AES-256-GCM being the default algorithm. There are no known major vulnerabilities and OpenVPN is considered secure. OpenVPN supports Perfect Forward Secrecy. Perfect forward secrecy means that the encryption key used to encrypt and decrypt data is changed automatically and regularly. If the encryption key is compromised, it exposes only a small portion of the

user's sensitive data. OVPN rotates encryption keys automatically every 45-75 minutes, ensuring consistent and constant security.

OpenVPN utilizza un protocollo di sicurezza personalizzato e SSL/TLS per lo scambio delle chiavi. Usa OpenSSL per la cifratura, dunque è disponibile un ampio numero di algoritmi di cifratura, in particolare basati su AES [BMM04, RFC3826]. Quello di default è AES-256-GCM, che garantisce un ottimo livello di sicurezza, specialmente riguardo confidenzialità, autenticazione dell'origine e integrità dei dati.

2.4.5 Autenticazione

A differenza della modalità Preshared Statick Key, la modalità TLS (preferita) usa il protocollo TLS per autenticare, instaurare una connessione sicura ed effettuare lo scambio delle chiavi simmetriche di sessione tra i peers. L'uso di TLS non solo offre un metodo automatico e sicuro per la distribuzione delle chiavi simmetriche, ma anche un modo per rinnovare tali chiavi in qualsiasi momento della comunicazione. Questo aspetto della modalità TLS offre ciò che è chiamato Perfect Forward Secrecy, che non è presente nella modalità PSK. I due step principali del protocollo TLS, a grandi linee, sono:

1. Negoziazione della connessione TLS: entrambi i lati della connessione si autenticano scambiandosi i certificati e verificando i certificati del lato opposto; se l'autenticazione ha successo, il protocollo procede allo step due; altrimenti, la connessione viene terminata.
2. Le chiavi di sessione sono negoziate attraverso il canale TLS sicuro appena stabilito.

2.4.6 Misure di sicurezza aggiuntive

OpenVPN offre diverse funzionalità di sicurezza: cifratura fino a 256-bit attraverso la libreria OpenSSL, anziché supportare IKE; lavora in user space, senza quindi necessità di effettuare operazioni sullo stack IP, e quindi operazioni kernel; ha la possibilità di far cadere i privilegi di root; entrare in una *chroot jail* dopo l'inizializzazione; applicare un

SELinux context dopo l'inizializzazione; offre supporto alle smartcard attraverso i token basati su PKCS 11.

2.4.7 Considerazioni

Si tratta di una soluzione per VPN matura e flessibile, con supporto a meccanismi di sicurezza all'altezza.

2.5 WireGuard

2.5.1 Panoramica

In IPsec, si ha una separazione netta tra il livello che si occupa dello scambio dati (IKE) e il livello di trasformazione (AH/ESP). Seppure sia una saggia separazione del punto di vista semantico, e decisamente corretta da un punto di vista di rete, ha lo svantaggio di aumentare la complessità implementativa. WireGuard, anziché implementare questa separazione, crea un'interfaccia di rete virtuale che può essere amministrata con le utility standard `ip` e `ifconfig`. Dopo aver configurato questa interfaccia con una chiave privata (e opzionalmente una PSK) e le varie chiavi pubbliche dei peers con cui dovrà comunicare in maniera sicura, la connessione è pronta ad essere instaurata. Scambio di chiavi, connessioni, disconnessioni e via dicendo avvengono dietro le quinte, e l'amministratore non deve configurare nessuno di questi aspetti. Le regole di firewalling possono essere configurate usando i tool standard, con la garanzia che i pacchetti che provengono da un'interfaccia di WireGuard saranno autenticati e cifrati. Per la sua semplicità, WireGuard è apparentemente meno incline a errori di configurazione rispetto ad IPsec.

WireGuard è in grado di instaurare esclusivamente tunnel di livello 3. Con questo approccio è infatti più semplice assicurare autenticità e origine dei pacchetti. Supporta sia IPv4 che IPv6 e può incapsulare sia v4-in-v6 che v6-in-v4.

WireGuard si concentra sulla semplicità e su una codebase facilmente ispezionabile, essendo allo stesso tempo estremamente performante e adatto a diversi ambienti. Combinando lo scambio di chiavi e la cifratura a livello 3 in un unico meccanismo e utilizzando un'interfaccia di rete virtuale anziché un livello di trasformazione, Wire-

Guard rompe con la tradizione per perseguire una soluzione ingegneristicamente solida apparentemente più pratica e sicura.

A partire dalla versione 5.6 del kernel Linux, WireGuard verrà incluso nel kernel stesso.

2.5.2 Protocolli utilizzati

Nell'implementazione di WireGuard, sono utilizzati i seguenti protocolli:

ChaCha20 per cifratura simmetrica, autenticata con Poly1305, utilizzando AEAD, come specificato in [NL15, RFC7539]

Curve25519 come Elliptic-curve Diffie-Hellman, un protocollo per la negoziazione delle chiavi

BLAKE2s per hashing e hashing con chiave, descritto in [SA15, RFC7693]

SipHash24 come chiavi per hashtable

HKDF come funzione per la derivazione delle chiavi, come spiegato in [KE10, RFC5869]

2.5.3 Cifratura

Dal punto di vista della cifratura utilizzata da WireGuard, rompe la tradizione delle altre soluzioni. Infatti, è intenzionalmente privo di flessibilità per quel che riguarda la scelta dei cifratori e dei protocolli utilizzati. Se vengono trovate falle in quelli scelti in fase di progettazione, tutti i terminali avranno bisogno di essere aggiornati. Come dimostrato dalla continua scoperta di vulnerabilità all'interno del protocollo TLS, dare la possibilità di scegliere quale cifrario usare aumenta enormemente la complessità.

2.5.4 Autenticazione

Per la distribuzione delle chiavi, WireGuard si ispira a OpenSSH, dove i due peers si scambiano le proprie chiavi pubbliche statiche. Il meccanismo con cui lo scambio avviene è basato sull'handshake `Noise IK` di Noise [Noi]. Dopo lo scambio delle chiavi, il peer che non ha iniziato la connessione deve aspettare ad usare la sessione fino a che non riceve un pacchetto cifrato dall'iniziatore, che dà conferma delle chiavi. Le

chiavi pubbliche sono lunghe 32 bytes e possono essere facilmente rappresentate con una codifica Base64 in 44 caratteri, che semplifica il trasferimento di esse attraverso vari mezzi. È supportata anche la Perfect Forward Secrecy, illustrata precedentemente.

2.5.5 Considerazioni

La semplicità di installazione è sicuramente un fattore che in ambienti piccoli ha la sua rilevanza; insieme alle prestazioni elevate, la rendono una soluzione da valutare al momento di installare un servizio VPN.

Capitolo 3

Realizzazione

3.1 Virtualizzatore

Un virtualizzatore è un software che si occupa di astrarre le risorse hardware di un computer/server, facendo da intermediario tra esse e il software che deve girarci sopra. Astraendo le risorse, è possibile distribuirle in maniera agile e ottimizzata tra i vari software che le richiedono. In ambito di virtualizzazioni server, la situazione più comune è la seguente: sulla macchina fisica è installato un **hypervisor**, che crea, gestisce e assegna risorse alle macchine virtuali, su cui viene installato un sistema operativo completo; sono poi queste macchine virtuali ad offrire effettivamente i servizi.

3.1.1 Caratteristiche e funzionamento

Gli hypervisor rendono la virtualizzazione possibile attraverso una traduzione delle richieste tra le risorse fisiche e quelle virtuali.

Fondamentalmente, ci sono due tipi di hypervisor: quelli detti **bare-metal**, che vengono eseguiti direttamente sull'hardware fisico e sono spesso installati allo stesso livello del BIOS sulla scheda madre, e quelli detti **in-hosting**, che girano come software standard su un sistema operativo (detto host system) installato sull'hardware fisico. Suddividendo le risorse, è possibile assegnarle non più a una sola macchina, ma a molteplici macchine virtuali.

Lo svantaggio degli hypervisor in-hosting è la loro maggiore latenza rispetto agli

hypervisor bare-metal. Ciò è dovuto al fatto che la comunicazione tra hardware e hypervisor non è diretta, ma deve passare attraverso il sistema operativo che lo ospita. Questo tipo di hypervisor è anche detto client hypervisor, essendo più comunemente utilizzato da utenti finali e per il testing di software, scenari in cui la latenza non è particolarmente rilevante.

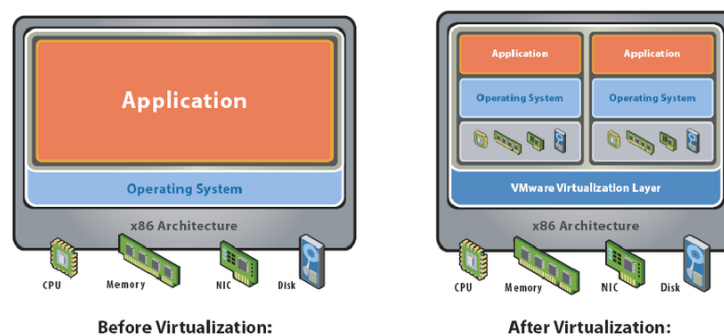


Figura 3.1: Installazione fisica vs Installazione virtuale

3.1.2 VirtualBox

Durante le prime fasi di testing, abbiamo utilizzato Oracle VirtualBox [Ora] sui nostri PC per creare un ambiente di simulazione. VirtualBox è un client hypervisor, open-source e disponibile per tutti i sistemi operativi.

3.1.3 VMWare ESXi

Nella fase di realizzazione, abbiamo invece utilizzato un server reale, su cui era installato l'hypervisor VMWare ESXi [vmw]. Una volta allocate le risorse necessarie, abbiamo installato CentOS 7 come sistema operativo sulla macchina virtuale come da requisito. Questa macchina virtuale corrisponde al router/firewall nel diagramma di rete presentato nei capitoli precedenti.

3.1.4 Installazione e configurazione dei servizi VPN

Dopo una fase iniziale di configurazione dei servizi interni ed esterni illustrati nei requisiti, si è passati all'installazione e configurazione dei tre servizi VPN scelti per il

testing.

3.2 Installazione e configurazione di IPsec - tunnelmode

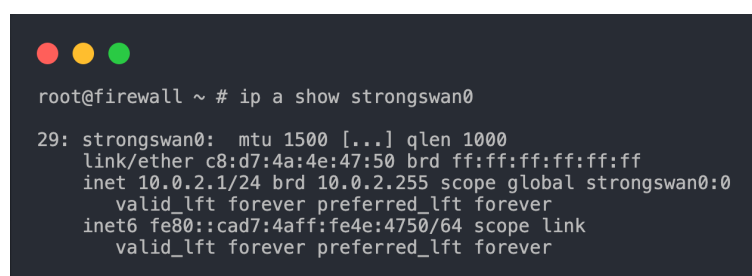
Il primo servizio che è stato installato è IPsec, grazie all'implementazione open-source offerta da strongSwan [str].

3.2.1 Aggiunta dell'interfaccia di rete virtuale

Prima di installare strongSwan, è necessario predisporre l'interfaccia di rete virtuale che verrà utilizzata per la creazione del tunnel VPN. Per fare ciò, è necessario abilitare le interfacce virtuali con il comando `modprobe dummy`. A questo punto è possibile aggiungerla e configurarla, assegnandole:

- un nome (e.g.: `strongswan0`)
- un MAC address qualunque - a patto che sia diverso da tutti quelli presenti nella subnet (e.g.: `C8:D7:4A:4E:47:50`),
- un indirizzo IP con la relativa maschera (e.g.: `10.0.2.1/24`)

e abilitarla. Una volta completato il tutto, visualizziamo il risultato con il seguente comando:



```
root@firewall ~ # ip a show strongswan0
29: strongswan0: mtu 1500 [...] qlen 1000
    link/ether c8:d7:4a:4e:47:50 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.1/24 brd 10.0.2.255 scope global strongswan0:0
        valid_lft forever preferred_lft forever
    inet6 fe80::cad7:4aff:fe4e:4750/64 scope link
        valid_lft forever preferred_lft forever
```

Figura 3.2: Dettagli interfaccia virtuale strongswan0

3.2.2 Installazione di strongSwan

Per installare il daemon di strongSwan è sufficiente lanciare il comando `dnf install strongswan`, che si occuperà di scaricare e installare la versione più recente disponibile.

3.2.3 Configurazione di strongSwan

Per adattare il comportamento del servizio alle proprie esigenze, è necessario modificare il file di configurazione del servizio, che nel caso di strongSwan si trova nella directory `/etc/strongswan/`. Al suo interno, andremo ad inserire i parametri adatti. Di particolare rilievo sono i seguenti:

```
conn strongswanVPN      # per la connessione di nome strongswan VPN
    type=tunnel          # IPSec in tunnel mode
    keyexchange=ikev2     # Protocollo per lo scambio chiavi

    # Algoritmi di cifratura consentiti
    ike=aes256-sha256-modp2048,[...],aes256gcm16-prfsha512-ecp384!
    esp=aes256-sha256-modp2048,[...],aes256gcm16-ecp384!

    leftcert=server.crt  # Certificato del server
    rightauth=eap-tls     # Protocollo di autenticazione
    rightdns=8.8.8.8      # DNS Server per il client

    # Range di indirizzi IP assegnabili ai client
    rightsourceip=10.0.2.10-10.0.2.100
```

3.2.4 Creazione del certificato per un client

Una volta terminata la configurazione lato server, è necessario creare un certificato per ogni client che vorrà connettersi alla VPN. È possibile farlo tramite le funzioni della libreria OpenSSL. Essendo in ambiente di testing e non di produzione, non ci si è preoccupati di acquistare un certificato che abbia possibilità di firmare altri certificati; si è scelto infatti di utilizzare sempre OpenSSL per creare una Certificate Authority locale. Questo ha lo svantaggio che i client non hanno modo di verificare presso un ente stabilito (e.g.: Let's Encrypt, o altre CA affermate) la validità del certificato, ma il funzionamento è identico.

3.3 Installazione e configurazione di OpenVPN over TCP

Per il servizio OpenVPN, si è deciso di utilizzare la porta 443 TCP per testare le performance nel caso di reti con firewall stringenti.

3.3.1 Installazione di openvpn

L'installazione avviene tramite il comando `dnf install openvpn`, che scarica e installa tutto il necessario per eseguire il servizio. Al file di configurazione di OpenVPN, sono state apportate alcune modifiche. Quelle più rilevanti sono le seguenti:

```
port 1194, proto tcp    # Porta e protocollo da utilizzare

# Dopo l'avvio, si disabilitino i privilegi di root
user nobody, group nobody

# Rende la VPN una sottorete, con prefisso e maschera
topology subnet
server 10.8.0.0 255.255.255.0

push "dhcp-option DNS 8.8.8.8"    # DNS primario per i clients
push "dhcp-option DNS 8.8.4.4"    # DNS secondario per i clients

# Comunica ai client di far passare il loro traffico
#     attraverso il server VPN
push "redirect-gateway def1 bypass-dhcp"

# Nomi dei file della CA, del certificato pubblico del server
#     e della relativa chiave privata
ca ca.crt
cert server_jH2NKwoak6pDEBoQ.crt
key server_jH2NKwoak6pDEBoQ.key
```

```
cipher AES-128-GCM # Algoritmo scelto per la cifratura del canale

tls-version-min 1.2 # Versione minima di TLS
tls-crypt tls-crypt.key
tls-cipher TLS-ECDHE-ECDSA-WITH-AES-128-GCM-SHA256
```

3.3.2 Certificati

Per la creazione della Certificate Authority e di tutti i certificati, sono state utilizzate le funzioni della libreria EasyRSA, messa a disposizione sulla repository GitHub di OpenVPN.

3.3.3 Configurazione del profilo VPN per un client

Affinché un client possa connettersi alla VPN, è necessario che egli sia in possesso di un profilo contenente tutti i dettagli tecnici necessari ad instaurare una connessione corretta con il server, e il suo certificato personale. In particolare, devono combaciare porta e protocollo utilizzato, gli algoritmi di cifratura, le funzioni di digest e i parametri per TLS.

3.4 Installazione e configurazione WireGuard

L'installazione di WireGuard si effettua semplicemente con il comando `dnf install wireguard`, che si generi la chiave del server e che si inserisca all'interno del file di configurazione, insieme all'IP che utilizzerà il server stesso per interfacciarsi con la VPN. In automatico viene creata un'interfaccia di rete virtuale `wg0`, a cui viene assegnato l'IP scelto. Ogni client avrà sempre lo stesso IP, in quanto vengono assegnati nella fase di creazione del profilo di connessione.

3.4.1 Configurazione del profilo VPN per un client

Affinché un client possa connettersi alla VPN, è necessario che nel file di configurazione del server sia presente la chiave pubblica del client, e nel profilo di connessione del client

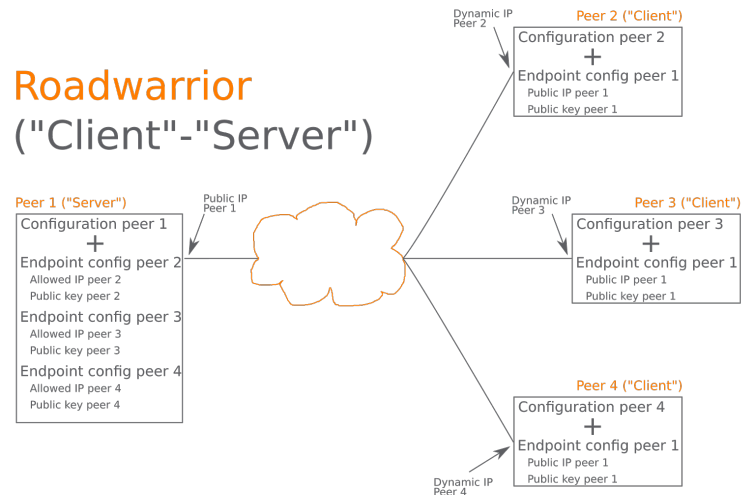


Figura 3.3: Architettura di installazione di WireGuard

sia presente l'IP pubblico del server, la sua chiave pubblica e l'indirizzo IP che il client assumerà.

3.5 Configurazione del Firewall

Nelle policies del firewall, sono state aggiunte delle voci che autorizzano il traffico proveniente dalle interfacce di rete virtuali `strongswan0`, `tun0` e `wg0` a navigare su Internet e a comunicare sulla porta 5201 TCP/UDP con la LAN dei server. Questa sarà la porta utilizzata da un servizio di misura delle performance che sarà in esecuzione sul Web Server, `iperf3`.

Capitolo 4

Testing

4.1 Modalità di esecuzione dei test

In questa sezione si andrà ad analizzare le performance delle tre soluzioni VPN descritte nei capitoli precedenti, al fine di valutare quale di esse offre le prestazioni migliori. Le prestazioni verranno valutate analizzando il throughput, la sua stabilità e la percentuale di packetloss. I software che sono stati utilizzati per effettuare le misurazioni questi dati sono `iPerf3` e `mtr`.

4.1.1 Panoramica di `iPerf3`

`iPerf` è uno strumento open source che permette di misurare le prestazioni di una rete. Per effettuare le misurazioni, `iPerf` crea dei flussi di dati su TCP, UDP o SCTP e invia traffico da un host all'altro; al termine del trasferimento, oltre a un report dettagliato in base al tipo di misurazione richiesta, mostra la larghezza di banda media disponibile. In questo modo, gli utenti possono determinare il throughput effettivamente utilizzabile.

4.1.2 Panoramica di `mtr`

`mtr` è un altro software di misurazione delle performance di una rete, che sostanzialmente unisce i risultati di `traceroute` e `ping`. I test effettuati da `mtr` sono unidirezionali, dunque è opportuno effettuare misurazioni manualmente in entrambe le direzioni, in

```

iperf3 -c firewall.filippovisconti.com -p 64999 -t 10s -i 1
Connecting to host firewall.filippovisconti.com, port 64999
[ 7] local 192.168.1.118 port 59050 connected to 79.61.138.204 port 64999
[ ID] Interval      Transfer    Bitrate
[ 7] 0.00-1.00 sec  7.29 MBytes 61.2 Mbits/sec
[ 7] 1.00-2.00 sec  5.75 MBytes 48.2 Mbits/sec
[ 7] 2.00-3.00 sec  5.64 MBytes 47.3 Mbits/sec
[ 7] 3.00-4.00 sec  5.61 MBytes 47.0 Mbits/sec
[ 7] 4.00-5.00 sec  5.59 MBytes 46.9 Mbits/sec
[ 7] 5.00-6.00 sec  5.57 MBytes 46.7 Mbits/sec
[ 7] 6.00-7.00 sec  5.57 MBytes 46.7 Mbits/sec
[ 7] 7.00-8.00 sec  5.57 MBytes 46.7 Mbits/sec
[ 7] 8.00-9.00 sec  5.57 MBytes 46.8 Mbits/sec
[ 7] 9.00-10.00 sec 5.59 MBytes 46.9 Mbits/sec
- - - - -
[ ID] Interval      Transfer    Bitrate
[ 7] 0.00-10.00 sec 57.7 MBytes 48.4 Mbits/sec
[ 7] 0.00-10.00 sec 56.9 MBytes 47.7 Mbits/sec
iperf Done.

```

Figura 4.1: Esempio di output di iPerf3

quanto i risultati differire in maniera sostanziale. Per avere una misurazione affidabile, è consigliabile far durare il test almeno 10 minuti.

```

> sudo mtr firewall.filippovisconti.com -s 5000 -c 10 -n -r
Start: 2022-06-28T16:18:24+0200
HOST: Filippos-MBP.lan
  1. |-- 192.168.1.254      0.0%  10    1.0    1.0    0.7    1.3    0.2
  2. |-- 10.103.123.42     0.0%  10    1.8   15.7    1.7  100.5   30.4
  3. |-- 10.103.11.38      0.0%  10    3.1    3.9    3.1    6.7    1.2
  4. |-- 10.1.170.1        0.0%  10    2.6    2.7    2.4    3.6    0.3
  5. |-- 10.254.2.2        0.0%  10    2.2    5.5    2.2   17.9    6.1
  6. |-- 89.97.200.190     0.0%  10    2.2    2.6    2.2    2.8    0.2
  7. |-- 89.97.200.61      0.0%  10    3.3    3.0    2.4    3.3    0.3
  8. |-- 85.36.8.152       0.0%  10    3.2    3.4    3.2    3.6    0.2
  9. |-- ???              100.0  10    0.0    0.0    0.0    0.0    0.0
 10. |-- ???              100.0  10    0.0    0.0    0.0    0.0    0.0
 11. |-- ???              100.0  10    0.0    0.0    0.0    0.0    0.0
 12. |-- 79.61.138.204    60.0%  10   12.3   12.1   11.8   12.4    0.3

```

Figura 4.2: Esempio di output di mtr

4.1.2.1 Spiegazione del formato dell'output

Nella prima colonna, si legge un numero e un indirizzo IP: il numero corrisponde alla distanza in hop tra l'host da cui parte il test e l'IP indicato alla sua destra. La seconda colonna, **Loss%**, indica la percentuale di pacchetti che quell'IP ha perso. È auspicabile un valore inferiore all'1% per una connessione affidabile. La terza colonna, **Snt**, indica il numero di pacchetti inviati a quell'IP. Le successive 4 colonne indicano il valore dell'ultimo, del medio, del migliore e del peggiore round-trip-time in millisencondi - ossia il tempo necessario affinché un pacchetto parta dal mittente, raggiunga il destinatario, e torni indietro. L'ultima colonna indica la deviazione standard tra questi ultimi 4 valori.

I valori dalla terza colonna in poi forniscono dunque informazioni sulla latenza della rete. È desiderabile il valore più basso possibile. Tuttavia, spesso la latenza dipende da fattori esterni alla rete locale.

Nella misurazione di esempio, è stato richiesto l'invio di 10 pacchetti (`-c 10`) di dimensione 5000 byte (`-s 5000`). Per gli hop 9, 10 e 11, si ha un risultato anomalo: nessun IP restituito e 100% di packet loss. Questo risultato non mostra problemi di connessione, ma indica semplicemente che l'host non ha risposto alle richieste indirizzate a lui (per i motivi più disparati, da un carico di lavoro troppo alto, a un firewall che fa cadere quel tipo di pacchetto), e che però, visto che l'hop 12 risponde correttamente, ha inoltrato correttamente quelle destinate a chi gli succede.

4.1.3 Scelta della configurazione di test

Ancora del testo

4.1.4 Criteri di valutazione

Per dare una valutazione complessiva alle tre soluzioni testate, si andranno a tenere in considerazione i seguenti parametri: throughput, percentuale di packet loss per un pacchetto di grandi dimensioni e latenza media.

4.1.4.1 Throughput

Il throughput di un canale di comunicazione misura la quantità di dati che può essere trasferita tra mittente e destinatario in una data unità di tempo. In ambito reti, si è soliti utilizzare come unità di tempo il secondo e come quantità di dati il bit, o suoi multipli (Kbit, Mbit, Gbit). La velocità e l'affidabilità di trasmissione dei pacchetti sono parametri fondamentali ed è necessario che siano in grado di soddisfare le necessità dell'azienda proprietaria della rete. Packet loss, latenza e jitter influenzano il throughput di una rete, e più sono elevati, più le performance degradano. Minimizzare tutti questi fattori è un punto cardine della progettazione e ottimizzazione di una rete. La larghezza di banda potrebbe essere confusa con il throughput; è un valore che misura sempre una quantità di bit trasferiti in un'unità di tempo, ma misura il limite massimo teorico, e non quello reale.

È importante sottolineare che una larghezza di banda maggiore non conferisce più velocità, bensì dà soltanto la possibilità di trasferire allo stesso momento una quantità di dati maggiore. Se si hanno problemi di latenza e di perdita dei pacchetti, questi non verranno risolti aumentando la larghezza di banda.

4.1.4.2 Packetloss

Quando un pacchetto non riesce a raggiungere la destinazione prevista, si verifica il fenomeno della perdita di pacchetti, packet loss. Un utente avverte questo problema come interruzioni della rete, perdita di connettività e una velocità di connessione rallentata. Le situazioni in cui si soffre maggiormente questo problema sono tutte quelle in cui è richiesta elaborazione di dati in real-time, dove i ritardi non sono tollerati.

The Transmission Control Protocol (TCP) detects packet loss and performs retransmissions to ensure reliable messaging. Packet loss in a TCP connection is also used to avoid congestion and thus produces an intentionally reduced throughput for the connection.

In real-time applications like streaming media or online games, packet loss can affect a user's quality of experience (QoE). Causes[edit] The Internet Protocol (IP) is designed according to the end-to-end principle as a best-effort delivery service, with the intention of keeping the logic routers must implement, as simple as possible. If the network made reliable delivery guarantees on its own, that would require store and forward infrastructure, where each router devotes a significant amount of storage space to packets while it waits to verify that the next node properly received them. A reliable network would not be able to maintain its delivery guarantees in the event of a router failure. Reliability is also not needed for all applications. For example, with live streaming media, it is more important to deliver recent packets quickly than to ensure that stale packets are eventually delivered. An application or user may also decide to retry an operation that is taking a long time, in which case another set of packets will be added to the burden of delivering the original set. Such a network might also need a command and control protocol for congestion management, adding even more complexity.

To avoid all of these problems, the Internet Protocol allows for routers to simply drop packets if the router or a network segment is too busy to deliver the data in a

timely fashion. This is not ideal for speedy and efficient transmission of data, and is not expected to happen in an uncongested network.[4] Dropping of packets acts as an implicit signal that the network is congested, and may cause senders to reduce the amount of bandwidth consumed, or attempt to find another path. For example, using perceived packet loss as feedback to discover congestion, the Transmission Control Protocol (TCP) is designed so that excessive packet loss will cause the sender to throttle back and stop flooding the bottleneck point with data.

Packets may also be dropped if the IPv4 header checksum or the Ethernet frame check sequence indicates the packet has been corrupted. Packet loss can also be caused by a packet drop attack.

Measurement[edit] Packet loss may be measured as frame loss rate defined as the percentage of frames that should have been forwarded by a network but were not.[8]

Acceptable packet loss[edit] Packet loss is closely associated with quality of service considerations. The amount of packet loss that is acceptable depends on the type of data being sent. For example, for voice over IP traffic, one commentator reckoned that "[m]issing one or two packets every now and then will not affect the quality of the conversation. Losses between 5

4.1.4.3 Latenza

Network latency:

The latency data is shown in the last five columns of the report (the second column shows the progressive number of the ICMP packets sent). Normally the latency increases proportionally to the test execution time. If the growth is proportional and does not present significant jumps, this means that there are no problems on the network. They do not indicate anomalies nor peak values on the single node 6.

Conversely, if from node 8 onwards high latency values are found (compared to the first 6 hops), which persist until the last hops, it could indicate the presence of numerous problems on the network, such as inadequate configurations of network cards or routers, abnormally working services or network congestion.

4.2 Misure senza VPN

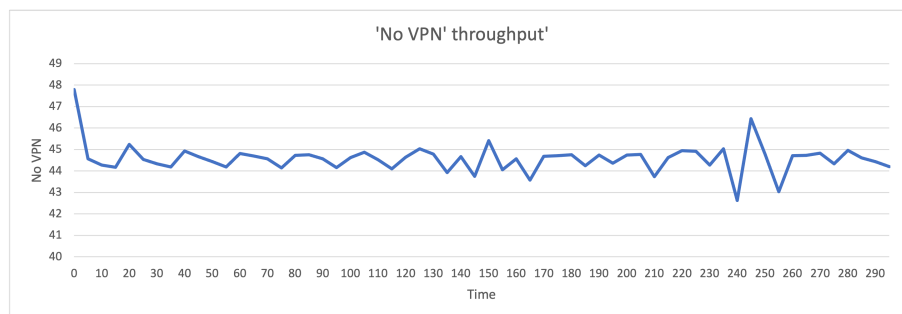


Figura 4.3: Throughput senza VPN su 300 secondi

4.3 Misure con IPSec e IKEv2

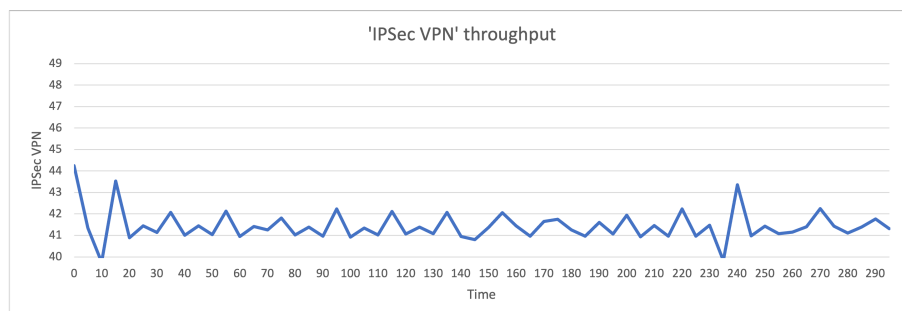


Figura 4.4: IPSec Throughput su 300 secondi

4.4 Misure con OpenVPN over TCP

4.5 Misure con WireGuard

4.6 Analisi delle misure

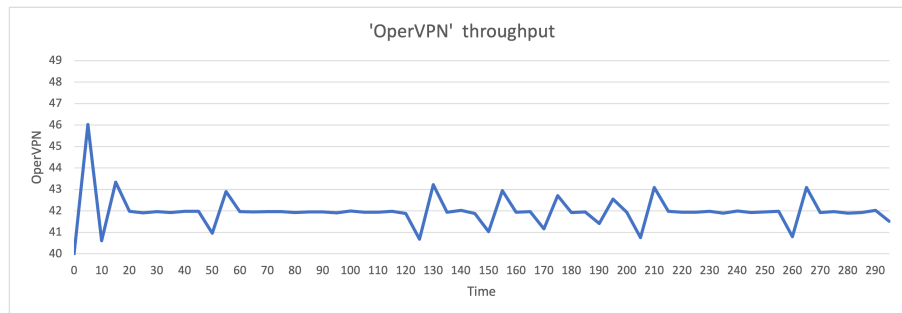


Figura 4.5: OpenVPN Throughput su 300 secondi

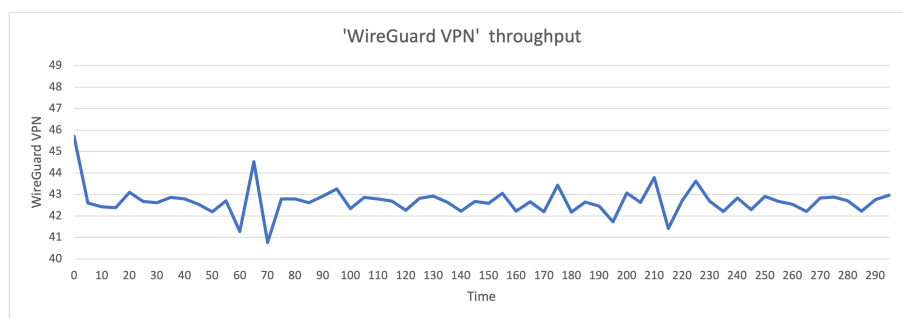


Figura 4.6: WireGuard Throughput su 300 secondi

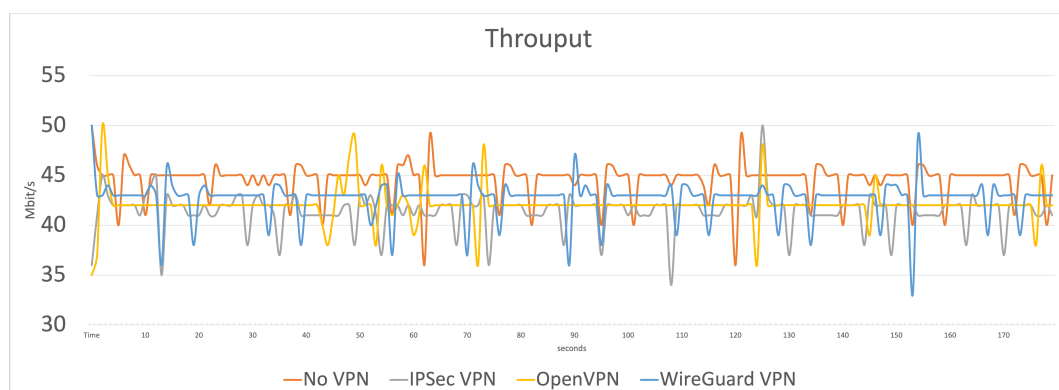


Figura 4.7: Confronto dei throughput grezzi

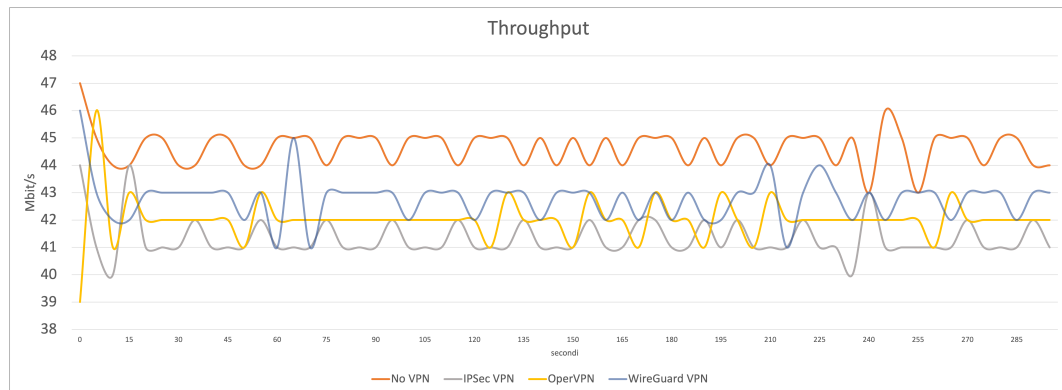


Figura 4.8: Confronto dei throughput raffinati

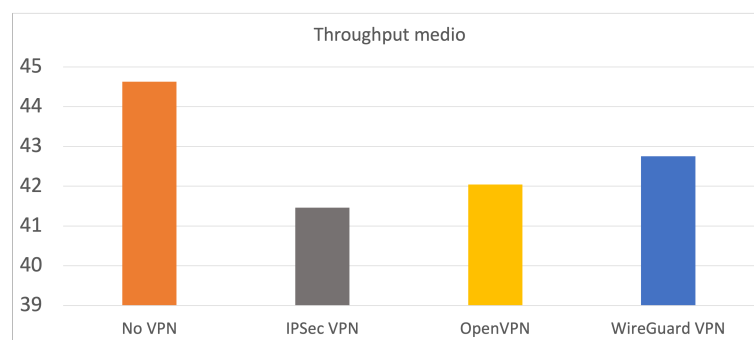


Figura 4.9: Confronto dei throughput medi

Capitolo 5

Security concerns

5.1 Principali problematiche di sicurezza

Ancora del testo

5.2 Attacchi mirati agli utenti

Ancora del testo

5.3 Attacchi mirati al sistema

Ancora del testo

5.4 Multi-factor authentication

Ancora del testo

5.4.0.1 Certificato

Ancora del testo

5.4.0.2 Username e password

Ancora del testo

5.4.0.3 One Time Password

Ancora del testo

Conclusioni e sviluppi futuri

Quale è uscito vincitore

Ancora del testo

Come migliorare le misure

Ancora del testo

Test di VPN Peer-To-Peer

Ancora del testo

Zero Tier

Ancora del testo

Bibliografia

- [Apa] Apache. <https://httpd.apache.org>.
- [BMM04] U. Blumenthal, F. Maino, and K. McCloghrie. The advanced encryption standard (aes) cipher algorithm in the snmp user-based security model. RFC 3826, RFC Editor, June 2004.
- [Cen] CentOS. <https://www.centos.org>.
- [CSF⁺08] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet x.509 public key infrastructure certificate and certificate revocation list (crl) profile. RFC 5280, RFC Editor, May 2008. <http://www.rfc-editor.org/rfc/rfc5280.txt>.
- [Dro97] Ralph Droms. Dynamic host configuration protocol. RFC 2131, RFC Editor, March 1997. <http://www.rfc-editor.org/rfc/rfc2131.txt>.
- [Fre00] N. Freed. Behavior of and requirements for internet firewalls. RFC 2979, RFC Editor, October 2000.
- [HPV⁺99] K. Hamzeh, G. Pall, W. Verthein, J. Taarud, W. Little, and G. Zorn. Point-to-point tunneling protocol (pptp). RFC 2637, RFC Editor, July 1999.
- [JS96] Trevor H. Jones and Il-Yeol Song. Analysis of binary/ternary cardinality combinations in entity-relationship modeling. *Data Knowledge Engineering*, 19(1):39–64, 1996.
- [Kau05] C. Kaufman. Internet key exchange (ikev2) protocol. RFC 4306, RFC Editor, December 2005. <http://www.rfc-editor.org/rfc/rfc4306.txt>.

- [KE10] H. Krawczyk and P. Eronen. Hmac-based extract-and-expand key derivation function (hkdf). RFC 5869, RFC Editor, May 2010. <http://www.rfc-editor.org/rfc/rfc5869.txt>.
- [Ken05a] S. Kent. Ip authentication header. RFC 4302, RFC Editor, December 2005. <http://www.rfc-editor.org/rfc/rfc4302.txt>.
- [Ken05b] S. Kent. Ip encapsulating security payload (esp). RFC 4303, RFC Editor, December 2005. <http://www.rfc-editor.org/rfc/rfc4303.txt>.
- [Lin] Red Hat Enterprise Linux. <https://www.redhat.com/en>.
- [Moc87] P. Mockapetris. Domain names - concepts and facilities. STD 13, RFC Editor, November 1987. <http://www.rfc-editor.org/rfc/rfc1034.txt>.
- [NL15] Y. Nir and A. Langley. Chacha20 and poly1305 for ietf protocols. RFC 7539, RFC Editor, May 2015. <http://www.rfc-editor.org/rfc/rfc7539.txt>.
- [Noi] Noise. <http://noiseprotocol.org/noise.pdf>.
- [Ora] Oracle. <https://www.virtualbox.org>.
- [Pos81] Jon Postel. Internet protocol. STD 5, RFC Editor, September 1981. <http://www.rfc-editor.org/rfc/rfc791.txt>.
- [PR85] J. Postel and J. Reynolds. File transfer protocol. STD 9, RFC Editor, October 1985. <http://www.rfc-editor.org/rfc/rfc959.txt>.
- [SA15] M-J. Saarinen and J-P. Aumasson. The blake2 cryptographic hash and message authentication code (mac). RFC 7693, RFC Editor, November 2015.
- [str] strongSwan. <https://www.strongswan.org>.
- [vmw] vmware. <https://www.vmware.com/it/products/esxi-and-esx.html>.