

UNIVERSITÀ POLITECNICA DELLE MARCHE

FACOLTÀ DI INGEGNERIA

Corso di Laurea Magistrale in Ingegneria Elettronica

Configurazione di architetture di reti carrier Configuration of carrier network architectures

Relatore: Prof. **Gambi Ennio** Tesi di laurea di: Alessio Antonucci

Correlatore: Prof. **De Santis Adelmo**

A.A. 2019/2020

Indice

1. Introduzione	3
2. Introduzione alle Enterprise Network	4
3. Topologia di rete	5
4. Protocolli per la configurazione	7
4.1. Introduzione al routing IP	7
4.2. IP static routes e OSPF	8
4.3. DHCP	9
4.4. GRE over IPsec	9
4.5. ICMP	0
5. eNSP	2
6. Wireshark 1	3
7. Configurazione della topologia di rete 1	4
7.1. Assegnazione degli spazi degli indirizzi1	4
7.2. Configurazione OSPF 1	6
7.3. Configurazione DHCP nelle reti Customer	23
7.4. Configurazione di percorsi predefiniti2	26
7.5. Configurazione del Tunnel GRE	32
7.6. Configurazione GRE over IPsec	37
8. Conclusione	12
Bibliografia4	13

1. Introduzione

In questa tesi si lavorerà in un ambito diventato ormai fondamentale al giorno d'oggi, cioè le reti Internet. Tutto viaggia in rete: informazioni, transazioni, voce, dati. Gestire la rete nel modo migliore, soprattutto se siamo nell'ambito aziendale, è ormai diventata una vera e propria necessità. Secondo Huawei Global Industry Vision (GIV), entro il 2025, ci saranno 100 miliardi di connessioni e 6,2 miliardi di persone che avranno accesso a Internet. Inoltre, 1'85% delle applicazioni aziendali sarà basato su cloud. A causa di tutto questo, le reti IP dovranno trasportare un numero crescente di servizi chiave. Per questi motivi, conoscere il funzionamento e avere la padronanza delle tecniche di routing sono fattori molto importanti, quanto come l'ottimizzazione della rete (cioè come il bilanciamento del traffico, la gestione della ridondanza, la stabilità della rete), la scalabilità oppure la gestione di LAN enterprise. Entrando più nello specifico del lavoro svolto, l'obbiettivo di questa tesi è la configurazione di una topologia di rete, che simula uno scenario reale in cui ci sono due reti customer interconnesse da una rete ISP (Internet Service Provider). I dispositivi di rete verranno interamente configurati per soddisfare le specifiche che verranno elencate più tardi all'interno della tesi. Per fare questo, si userà il software eNSP fornito da Huawei, grazie al quale sarà possibile simulare la rete in questione.

2. Introduzione alle Enterprise Network

I requisiti per un'azienda evidenziano la necessità di avere reti in grado di adattarsi alle esigenze in continua evoluzione in termini di business e servizi. Occorre introdurre alcuni principi di funzionamento di una rete aziendale e descrivere come è formata e si adatta alle varie esigenze del mondo reale. L'introduzione di Internet come dominio di rete pubblica ha portato ad una estensione della rete aziendale, attraverso la quale possono essere collegate altre reti in un altro punto geografico appartenenti a una singola organizzazione o entità (nel progetto trattato in questa tesi, si vedrà un esempio di quanto è stato appena detto). Ciò porta una serie di nuove sfide per stabilire la connettività tra queste reti, dovendo anche mantenere la privacy e la sicurezza dei dati appartenenti alla singola organizzazione. La rete IP pubblica e di terze parti consentono queste interconnessioni, possibili anche grazie allo sviluppo di tecnologie che stabiliscono connessioni di reti private su infrastrutture pubbliche. Nelle *Large Enterprise* l'architettura di rete è divisa in più livelli:

- Livello di accesso, di cui fanno parte apparati con una bassa capacità di elaborazione, ma che devono avere una grande densità di porte siccome gli utenti saranno collegati a questi apparati.
- Livello di aggregation, che consente di collegare fra loro tanti apparati di tipo access.
 A questo livello ho traffico elevato (dovuto anche al fatto della forte ridondanza), perciò serviranno dispositivi performanti.
- Livello di core, dove abbiamo prevalentemente dispositivi di livello 3. Quindi si introduce la funzione di instradamento del traffico (connettività verso global internet).
- Server farm, cioè dispositivi di rete che erogano servizi verso l'azienda oppure verso l'esterno.

Molto più spesso, però, abbiamo a che fare con un'architettura *Small Enterprise* in cui alcuni dei livelli sopra elencati sono collassati in uno. Questo perché realizzare per intero l'architettura descritta prima è molto costoso, soprattutto a causa della ridondanza (utile per avere una rete più affidabile e robusta), della scalabilità (per avere una possibilità di espansione futura) e della modularità (un problema nella rete non si deve riflettere su tutta la struttura, ma solo in una sua parte).

3. Topologia di rete

La figura 1 mostra la topologia di rete da configurare. Abbiamo due semplici reti customer, che potranno comunicare attraverso la rete ISP formata dai router R1, R2, R3, R4, R5, R10. Possiamo notare che la rete ISP fornisce ridondanza (comunque non eccessiva), così che il funzionamento non venga compromesso a causa di un guasto di un router o di un collegamento, come verrà mostrato successivamente in alcuni casi nei prossimi capitoli. Comunque, non è difficile trovare dei punti critici in questa rete sotto il punto di vista dell'affidabilità. Innanzitutto, non ci sono collegamenti di backup tra i router, caratteristica che non rappresenta un gran problema per i router più interni della rete poiché hanno altri percorsi alternativi possibili, ma è un problema per il collegamento tra il router CE (Customer Edge) e il router PE (Provider Edge). Se, per esempio, il collegamento tra R8 e R3 si interrompe o se uno dei due router smette di funzionare, allora la rete Customer 1 non potrà connettersi alla rete e quindi non sarà raggiungibile e rimarrà isolata. In altre parole, R8 e R3 costituiscono un *single-point-of-failure* nel collegamento di Customer 1 alla rete. Stesso discorso vale per la rete Customer 2 con i router R5 e R9.



Figura 1. Topologia di rete da configurare.

Ora qui di seguito elencheremo gli obbiettivi di questo progetto, cioè i passi per la configurazione della rete, che saranno svolti nell'ordine indicato:

- Assegnare degli spazi di indirizzi /30 a tutti i link tra i router;
- Configurare OSPF nella rete;
- Configurare DHCP nelle reti dei customer;
- Garantire la raggiungibilità tra la rete del customer 1 e del customer 2;

- ISP deve garantire che il percorso R3 R4 R5 sia sempre libero, in quanto dedicato a clienti enterprise
- Configurare la rete in modo che R9 ed R8 scambino pacchetti sul path R3 R1 R2 R5 e, in caso di fail: R3 R10 R5. Verificare il corretto funzionamento della rete.
- A seguito di una riorganizzazione aziendale, customer 1 e customer 2 si sono uniti in una sola azienda: implementare un tunnel GRE che consenta la comunicazione tra gli spazi di indirizzi dei customer (tra R8 ed R9), senza interessare ISP. Verificare con tracert il corretto percorso dei pacchetti.
- Dopo alcuni mesi di servizio, le aziende decidono di implementare una maggiore sicurezza nel collegamento. Configurare IPSec tra R9 ed R8. Veicolare GRE over IPSec.

4. Protocolli per la configurazione

In questo capitolo si introdurranno alcuni concetti e protocolli utili per capire il funzionamento della rete e come realizzarla. Nello specifico, si descriverà il principio di funzionamento del routing IP, il caso statico (manuale) per la realizzazione dei percorsi da inserire nelle tabelle di routing dei router e il caso dinamico attraverso il protocollo OSPF, l'assegnazione automatica degli indirizzi IP attraverso il protocollo DHCP, l'implementazione della sicurezza nella rete grazie all'uso combinato dei due protocolli GRE e IPsec e, infine, il testing della rete attraverso le operazioni di *ping* e *traceroute* eseguibili grazie al protocollo ICMP.

4.1. Introduzione al routing IP

Un dominio di rete è generalmente costituito da più reti. Per questo motivo, sono necessarie decisioni di instradamento (routing) per garantire che vengano utilizzate rotte ottimali per l'inoltro di pacchetti IP alle destinazioni di rete previste. Questo compito spetta al router, che prenderà delle decisioni seguendo dei criteri precisi. Un router ha una tabella di routing, in cui ci sono tutti gli spazi di indirizzi che conosce e sa come instradare, e una FIB (Forward Information Base), dove il router inserisce i percorsi preferiti per guidare l'inoltro dei pacchetti. Se il pacchetto IP ricevuto è indirizzato verso uno spazio degli indirizzi sconosciuto, il router scarterà il pacchetto oppure, se configurato, lo invierà ad un altro router (*default gateway*) cedendogli il compito di prendere una decisione su questo pacchetto. Le rotte nella tabella di routing vengono inserite guardando i seguenti parametri, in questo preciso ordine (si guarda il parametro successivo in caso di parità per risolvere le contese):

- Longest match: si guarda la rete di destinazione del pacchetto e la si confronta con gli spazi degli indirizzi presenti nel FIB del router. Chi avrà una corrispondenza maggiore, in termini di numero di bit, verrà scelto come percorso ottimale per inoltrare quel pacchetto.
- Preference: questo valore specifica la "fiducia" associata alla fonte di informazioni per la popolazione della tabella di routing. Viene utilizzato per decidere quale protocollo viene applicato alla tabella di routing in cui due protocolli offrono rotte simili. Un valore di preferenza più basso, indica una priorità maggiore.
- **Cost:** se non è possibile distinguere il percorso in base al *longest match* o alla *preference*, la metrica del costo viene presa in considerazione per identificare il percorso

che deve essere inserito nella tabella di routing. Ogni segmento ha un costo, che viene valorizzato sulla base della metrica del protocollo di routing utilizzato. Combinando questi segmenti per formare un percorso verso una destinazione, si ottiene un costo di percorso. Il percorso con il costo minore definisce la rotta ottimale per la destinazione specificata.

4.2. IP static routes e OSPF

L'aggiunta di rotte all'interno della tabella di routing può essere definita manualmente tramite *static routes* oppure mediante l'uso di *protocolli di routing dinamici*.

Lo *static route* consente il controllo diretto sulla tabella di routing, tuttavia, la configurazione manuale viene spesso utilizzata come complemento ai protocolli di routing dinamici per fornire rotte alternative o percorsi predefiniti di backup (nella configurazione della nostra topologia, infatti, avrà questo ruolo di supporto). Lo svantaggio delle routes statiche è che non possono adattarsi automaticamente ai cambiamenti della rete. Bisognerebbe, in caso di modifica topologica, riconfigurarle tutte manualmente. Per questo motivo, i percorsi statici sono adatti per reti con topologie semplici, poiché, in una struttura più complessa, il lavoro di riconfigurazione sarebbe eccessivo. Tuttavia, l'implementazione manuale delle rotte riduce il consumo di risorse nella rete, poiché i dispositivi devono eseguire meno operazioni.

OSPF è un protocollo di routing dinamico di tipo *link-state* progettato per le reti IP. La sua applicazione come IGP (Interior Gateway Protocol) all'interno di reti di grandi dimensioni porta numerosi vantaggi. Il funzionamento dei protocolli link-state può essere suddiviso in tre fasi:

- Network Discovery: ricerca dei router vicini e analisi delle informazioni in loro possesso.
- **Topology Database Exchange:** i router collaborano per avere tutti la stessa visione della topologia.
- Route Computation: ogni router analizza le informazioni in suo possesso e quelle ottenute dagli altri router per determinare il percorso migliore verso ogni spazio degli indirizzi. Da notare che la scelta di questo best-path è indipendente dalle scelte degli altri router.

OSPF è in grado di rilevare rapidamente le modifiche topologiche all'interno di una rete e di stabilire rotte libere da loop, con un overhead di comunicazione minimo per la negoziazione delle nuove rotte (si scambiano soltanto le informazioni strettamente necessarie). OSPF fornisce una soluzione anche per il problema della *scalabilità*, dando la possibilità di dividere la rete in aree, così da ridurre il sovraccarico di informazioni scambiate tra i router in reti molto grandi. La convergenza del processo OSPF richiede che ogni router sia a conoscenza dello stato delle sue interfacce e di quelle dei suoi vicini, in modo che sia possibile stabilire il percorso migliore verso ogni spazio di indirizzi. Questo si forma attraverso il flooding di LSA (Link State Advertisement), che sono unità dati che "pubblicizzano" reti note e stati di collegamento per ciascuna interfaccia all'interno del dominio di routing. Alla fine della convergenza, ogni router avrà un LSDB (Link State DataBase) che fornisce le basi per stabilire il percorso più breve per ciascun spazio degli indirizzi, le cui rotte vengono messe nella tabella di routing.

4.3. DHCP

DHCP è un protocollo per automatizzare l'assegnazione di indirizzi IP. DHCP assume rilevanza quando siamo di fronte a un numero considerevole di host, ognuno dei quali necessita una configurazione per partecipare alla rete IP. La configurazione manuale di tutti gli host richiede tanto tempo, ma soprattutto può essere fonte di malfunzionamenti nella rete a causa di errori umani, come ad esempio la duplicazione degli indirizzi o errori sulla configurazione. DHCP è perfetto per garantire la corretta assegnazione degli indirizzi e per ridurre l'onere amministrativo. È altrettanto utile nei casi in cui lo spazio degli indirizzi non riesca a coprire interamente il numero di nodi della rete. Se i nodi hanno basso coefficiente di contemporaneità, questi possono negoziare dinamicamente con il DHCP server l'assegnazione di un indirizzo IP temporaneo. Da tutto questo, si può dedurre quanto questo protocollo sia essenziale in una rete di grande dimensione.

4.4. GRE over IPsec

L'applicazione congiunta dei protocolli GRE e IPsec serve per sopperire le loro mancanze. *GRE*, fornisce l'incapsulamento dei pacchetti di un protocollo in pacchetti di un altro protocollo e crea un tunnel virtuale tra due router in modo da far passare dei pacchetti dati attraverso una rete che non li supporta (qualsiasi router posto tra i due estremi del tunnel non processeranno i pacchetti incapsulati). Uno dei limiti di GRE non fornisce protezione ai pacchetti mentre questi

vengono trasportati sulla rete pubblica, poiché non ha funzioni per la sicurezza. È per questo motivo che *IPsec* viene utilizzato insieme a GRE, così da consentire la creazione del tunnel GRE all'interno del tunnel IPsec per fornire le funzionalità di integrità e sicurezza (tramite crittografia) a GRE. IPsec utilizza due protocolli per fornire sicurezza al traffico: *AH* (*Authentication Header*) e *ESP* (*Encapsulating Security Payload*). AH permette di avere integrità connectionless, autenticazione dell'origine dei dati e protezione contro attacchi replay. ESP consente di avere la confidenzialità del traffico tramite la crittografia e opzionalmente anche l'autenticazione. IPsec ha due modalità di lavoro: *Transport Mode*, che fornisce principalmente protezione ai protocolli di livello superiore, e *Tunnel Mode*, che garantisce confidenzialità al traffico incapsulando il pacchetto che si vuole proteggere in un altro pacchetto IP (*tunneling*). Quindi, IPsec consente la comunicazione sicura tra reti private remote anche durante il passaggio nella rete pubblica, però non consente il trasporto di protocolli diversi da IPv4. È in questo momento che entra in gioco GRE per sopperire a tale mancanza di IPsec, supportando il trasporto di qualsiasi protocollo.

Il *Tunnel* è un canale di comunicazione che collega direttamente due nodi, a prescindere della topologia fisica che li separa. I nodi agli estremi del tunnel pensano di essere direttamente collegati tra loro. Quando un nodo riceve un pacchetto, analizza il suo header IP e determina l'indirizzo di destinazione. Questo viene confrontato con la tabella di routing per capire su quale interfaccia instradare il pacchetto. Se l'interfaccia è di tipo Tunnel (interfaccia logica), il pacchetto viene incapsulato all'interno di un altro pacchetto IP che ha come indirizzo di destinazione l'altro estremo del Tunnel. Il pacchetto originale non ha idea dell'effettivo percorso che sarà seguito ed un osservatore esterno vedrà solamente un pacchetto IP (quello scambiato tra i tunnel) che contiene dei dati.

4.5. ICMP

ICMP è un protocollo che lavora insieme a IP (è incapsulato al suo interno) e serve per compensare la limitata affidabilità del protocollo IP. ICMP è progettato per trasmettere messaggi di notifica, segnalazioni di errore e per informazioni diagnostiche. Lo scopo di questi messaggi è fornire un feedback su eventuali problemi presenti nella rete. Le principali operazioni di ICMP che ci interessano in questa tesi sono il *ping* e il *traceroute*. Il *ping* può essere utilizzato come strumento per verificare la raggiungibilità di un dispositivo nella rete e per raccogliere altre informazioni correlate, come il tempo di andata e ritorno dell'*Echo Request* e l'*Echo Reply* oppure la percentuale di pacchetti persi. Il *traceroute* serve per visualizzare i

nodi di rete che il pacchetto IP attraversa per raggiungere la propria destinazione e il ritardo hop-by-hop. Per fare questo, si imposta inizialmente il TTL (Time To Live) del pacchetto a 1 e lo si incrementa finché il pacchetto non raggiungerà la destinazione. Ogni volta che il TTL scadrà, sarà generato un messaggio ICMP dal nodo in cui è arrivato il pacchetto verso la sorgente, consentendo una valutazione hop-by-hop del percorso del pacchetto, utile per identificare punti di perdita, il ritardo nella rete e per scoprire loop di routing.

5. eNSP

Enterprise Network Simulation Platform (eNSP) è una piattaforma di simulazione di rete grafica, gratuita ed estensibile sviluppata da Huawei e rappresenta l'ambiente nel quale è stato sviluppato il presente lavoro di tesi. Simulando router e switch Enterprise Huawei, il software può creare vari scenari e può simulare reti di grandi dimensioni. Gli utenti possono quindi eseguire dei test e apprendere le tecnologie di rete senza utilizzare dispositivi reali. Questo software di simulazione ha un ruolo importante nell'apprendimento delle tecnologie emergenti e nella modellazione di reti, poiché permette una comprensione accurata del comportamento e delle prestazioni della topologia o della scalabilità di dispositivi, collegamenti e applicazioni. Elenchiamo qui di seguito i vantaggi nell'usare eNSP:

- Velocizza l'apprendimento: consente di comprendere e padroneggiare il funzionamento, la configurazione e l'ottimizzazione dei prodotti correlati in un ambiente virtuale.
- Ottimizzazione della rete: Rispecchiando accuratamente il comportamento della rete fisica, fornisce informazioni approfondite sulle prestazioni in tempo reale dei dispositivi di rete, collegamenti, protocolli e traffico. Consente di identificare rapidamente i problemi e mettere a punto i componenti per migliorare l'efficienza e le prestazioni della rete. Questa visione globale permette la pianificazione, costruzione, funzionamento e manutenzione per costruire topologie efficienti.
- Valutazione della cyber-resilience: fornisce la base per testare la resilienza informatica della rete.
- **Testing di applicazioni e servizi**: fornisce un ambiente per valutare le prestazioni in una grande varietà di condizioni.

6. Wireshark

Uno strumento molto potente per la comprensione del funzionamento della rete e dei vari protocolli è Wireshark. Si tratta di un software per l'analisi di protocolli che cattura i pacchetti (*packet sniffer*) in transito nella rete e ne rende possibile l'analisi con un elevato dettaglio. L'accesso a tali informazioni permette il rivelamento e la risoluzione di problemi, di effettuare un'analisi molto approfondita su ogni parte del pacchetto, in modo da avere una comprensione molto particolare e minuziosa del funzionamento della rete e il controllo delle attività presenti in essa. Inoltre, permette di visualizzare sequenzialmente l'intero flusso di traffico, dando anche la possibilità di filtrare quello che ci interessa specificando il tipo di protocollo, i numeri di porta, gli indirizzi di sorgente e destinazione oppure altri parametri dei vari layer di protocollo del pacchetto. Possiamo vedere un esempio del funzionamento di Wireshark dalla figura 2, in cui è possibile notare l'uso di un filtro, per selezionare solo un tipo di traffico e tutte le informazioni all'interno di un pacchetto di rete.

🚄 *Wi-Fi			_	o ×			
File Modifica Visualizza Vai Cattura Analizza	Statistiche Telefonia Wireless	Strumenti Aiuto					
🖌 🔲 🧷 🔘 📙 📇 🕅 🔂 🍳 🚗 🔿 🗺 🕡							
(dns							
No. Time Source	Destination Protocol	Length Info					
123 9.588687 192.168.1.30	192.168.1.1 DNS	73 Standard query 0x3357 A www.google.it					
124 9.589615 192.168.1.30	192.168.1.1 DNS	75 Standard query 0x9720 A www.gstatic.com					
128 9.609493 192.168.1.1	192.168.1.30 DNS	89 Standard query response 0x3357 A www.goog	le.it A 142.250.184.99				
129 9.612002 192.168.1.1	192.168.1.30 DNS	91 Standard query response 0x9720 A www.gsta	tic.com A 142.250.184.99				
→ 365 11.923306 192.168.1.30	192.168.1.1 DNS	73 Standard query 0x5bb9 A www.univpm.it					
← 374 11.948882 192.168.1.1	192.168.1.30 DNS	89 Standard query response 0x5bb9 A www.univ	pm.1t A 193.205.131.122				
415 12.114379 192.168.1.30	192.168.1.1 DNS	72 Standard query 0x7631 A www.bing.com					
416 12.136301 192.168.1.1	192.168.1.30 DNS	220 Standard query response 0x7631 A www.bing	.com CNAME a-0001.a-afdentry.net.trafficmanager.net CNAME www	w-bing-com.du			
486 12.877394 192.168.1.30	192.168.1.1 DNS	84 Standard query 0xd570 A www.google-analyt	ics.com				
488 12.900548 192.168.1.1	192.168.1.30 DNS	144 Standard query response 0xd5/0 A www.goog	le-analytics.com CNAME www-google-analytics.l.google.com A 21	16.58.208.142			
612 16.119949 192.168.1.30	192.168.1.1 DNS	93 Standard query 0x4/dc A livetileedge.dsx.	mp.microsoft.com				
613 16.14154/ 192.168.1.1	192.168.1.30 DNS	252 Standard query response 0x4/dc A livetile	edge.dsx.mp.microsoft.com UNAME livetileedge.xbetservices.aka	adns.net CNAM			
<pre>> Frame 365: 73 bytes on wire (584 bits), 75 bytes captured (584 bits) on interface \Device\UPF(04DE6F00-90E4-4DC5-8991-9CF90AF86532), id 0 > Ethernet II, Src: Hondbarp.abit52c (Dbit: Echnico_69:71:66 (10:13:31:69:71:66) * Internet Protocol Version 4, Src: 192.168.1.30, Dbit 192.168.1.1 0100 e Version: 4 0101 = Header Length: 20 bytes (5) > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT) Total Length: 59 Identification: 0xc007 (49159) > Flags: 0x0000 0 0000 0000 e= Fragment offset: 0 Time to live: 128 Protocol: 0UP (17)</pre>							
[Header checksum status: Unverified] Source: 192.168.1.30 Destination: 192.168.1.1 > User Datagram Protocol, Scc Port: 57552, > Domain Name System (query) 0000 101 33 169 71 6e b0 10 41 ab 15 2c 0010 00 3b c0 07 00 00 80 11 f7 3 ac 0 a8 0010 01 01 ab d0 A0 35 ao 27 ff bh 55 b0	Dst Port: 53	··E·		^			
0030 00 00 00 00 00 00 00 03 77 77 77 06 75	6e 69 76 70 ·····	nivp		~			
🔘 🍸 Domain Name System: Protocol			Pacchetti: 664 · visualizzati: 12 (1.8%) · scartati: 0 (0.0%)	Profilo: Default			

Figura 2. Esempio di cattura su Wireshark.

7. Configurazione della topologia di rete

Dopo aver introdotto e descritto i protocolli e gli strumenti utili per il progetto, si può procedere con la configurazione della topologia di rete presentata nel capitolo 3. In questo capitolo vedremo la configurazione passo per passo (seguendo l'ordine indicato) e verificheremo il corretto funzionamento alla fine di ogni operazione.

7.1. Assegnazione degli spazi degli indirizzi

Come prima cosa ci viene richiesto di assegnare ad ogni interfaccia dei router presenti nella rete uno spazio degli indirizzi. Inoltre, si specifica che i link tra i router devono avere uno spazio degli indirizzi /30, che rappresenta il prefix length, cioè il numero di bit presenti nell'indirizzo IP che formano il campo network più il campo subnet. Prima di mostrare il processo di configurazione, facciamo delle brevi considerazioni su quale classe di indirizzi implementare, siccome non viene specificato. Innanzitutto, sappiamo che N+S=30 bit (N sta per campo Network e S per campo Subnet) e, lavorando con indirizzi IPv4 (lunghi 32 bit), H=32-(N+S)=2 bit (H sta per campo Host). Avendo 2 bit per il campo Host, abbiamo 2²=4 indirizzi possibili, di cui soltanto 2 sono disponibili per l'assegnazione, poiché un indirizzo è dedicato al Network Address (tutti i bit del campo Host a 0) e l'altro al Broadcast Address (tutti i bit del campo Host a 1). Il primo serve per rappresentare una determinata rete, mentre il secondo serve per eseguire una trasmissione a tutte le destinazioni possibili presenti in una determinata rete. Ora dobbiamo stabilire la grandezza dei campi Network e Subnet. Si è scelto di usare gli indirizzi appartenenti alla classe C (tutti gli indirizzi che iniziano con 110), cioè con N=24 bit. In questo modo abbiamo che S=32-(N+H)=6 bit, avendo così 2⁶=64 sottoreti possibili. La scelta della classe C è stata fatta per ridurre il numero di bit usati dal campo Subnet, poiché ogni volta che si crea una sottorete, questa sarà caratterizzata da un Subnet Address ed un Subnet Broadcast Address. Per questo motivo, per ogni subnet che creiamo, perdiamo due indirizzi che potevano essere assegnati agli host. Quindi, fare subnetting molto aggressivo rischia di risultare uno spreco di risorse. Sempre per queste considerazioni, si è deciso di utilizzare un classe C /24 "puro" (cioè senza subnet) nel link tra host e il router CE. Come ultima cosa prima di passare a eNSP, gli indirizzi assegnati sono stati scelti secondo questo criterio per facilitare memorizzazione di tutti gli spazi di indirizzi: prendendo per esempio il link tra i router R8 (a sinistra del collegamento)

e R3 (a destra del collegamento), la rete sarà 192.168.**83**.0 e chi starà a sinistra del collegamento avrà come ultimo ottetto dell'indirizzo il valore decimale .1, mentre chi starà a destra .2.

Passiamo alla configurazione della topologia su eNSP. Dopo aver dato un nome ad ogni router attraverso il comando *sysname* nella view di sistema, quello che si andrà a fare è un lavoro semplice, ma abbastanza lungo, che consiste nell'assegnare un indirizzo IP ad ogni interfaccia di ogni router della rete seguendo la specifica e i criteri descritti prima. Per mostrare un esempio di configurazione, prendiamo il caso di R8 [figura 3].

E AR8			
AR8			
The device is running!			
<huawei>undo terminal monitor</huawei>			
Info: Current terminal monitor is	oII.		
<huawei>sys</huawei>			
Enter system view, return user vi	ew with Ctrl+Z.		
[Huawei]sysname R8			
[R8]int gig 0/0/0	100 100 00 1 00		
[R8-GigabitEthernet0/0/0]ip addre	ss 192.168.83.1 30		
[R8-GigabitEthernet0/0/0]int gig	0/0/1		
[R8-GigabitEthernetU/U/I]ip addre	ss 192.168.18.2 24		
[R8-GigabitEthernet0/0/1]quit			
[R8]dis ip interface brief			
*down: administratively down			
down: standby			
(1): loopback			
(s): spooling			
The number of interface that is U	P in Physical is 3		
The number of interface that is D	OWN in Physical is I		
The number of interface that is U	P in Protocol is 3		
The number of interface that is D	OWN in Protocol is 1		
Interface	IP Address/Mask	Physical	Protocol
GigabitEthernet0/0/0	192.168.83.1/30	up	up
GigabitEthernet0/0/1	192.168.18.2/24	up	up
GigabitEthernet0/0/2	unassigned	down	down
NULLO	unassigned	up	up(s)

Figura 3. Configurazione degli indirizzi IP sulle interfacce e visualizzazione dei risultati.

Il comando chiave per questo passo della traccia è *ip address*, che viene eseguito nella view di interfaccia e che serve per assegnare a quella determinata interfaccia l'indirizzo IP specificando anche il *prefix length* (da notare che dalla parte dell'host abbiamo un /24 e dalla parte della rete ISP abbiamo un /30). Attraverso il comando *display ip interface brief*, possiamo visualizzare tutte le interfacce che sono state configurate insieme ad alcune loro caratteristiche.

Nella figura 4 mostriamo la topologia con tutti gli indirizzi annotati che sono stati configurati.



Figura 4. Indirizzi IP nella topologia di rete dopo la configurazione.

Per verificare la raggiungibilità dei router nei collegamenti diretti, possiamo usare il comando di *ping* dai vari nodi della rete. Per esempio, verifichiamo la raggiungibilità di R3 da R8 [figura 5].



Figura 5. Ping tra R8 e R9.

L'*Echo Request* di R8 ha avuto risposta da R3 con un *Echo Reply*, confermando la raggiungibilità tra i due router e fornendo le statistiche descritte nel paragrafo 4.5.

7.2. Configurazione OSPF

Prima di procedere alla configurazione del processo OSPF, occorre fare un'altra operazione preliminare per ottimizzare le prestazioni della rete che vuole usare questo protocollo di routing dinamico. Il tipo di collegamento *Ethernet* viene visto come un mezzo di tipo broadcast dal protocollo OSPF, facendo partire l'elezione per il Designated Router (DR). Quando ho dei collegamenti punto-punto con supporto fisico Ethernet (come nel caso di tutti i link presenti nella nostra topologia), si verificheranno elezioni DR non necessarie, che generano un traffico

eccessivo. Per questo motivo, si è scelto di configurare su ogni interfaccia di ogni router il tipo di rete punto-punto attraverso il comando *ospf network-type p2p* eseguibile nella view di interfaccia. In questo modo, non si effettueranno le comunicazioni per eleggere il DR, così da non sprecare le risorse della rete.

Una volta fatto questo, si può procedere alla configurazione OSPF sull'intera rete. Anche questo lavoro richiede abbastanza tempo e attenzione, siccome per ogni router dovremo creare il processo OSPF a cui farà parte e abilitare il processo su ogni interfaccia, così che possa dichiarare di conoscere il relativo spazio degli indirizzi. Se tutto è impostato correttamente, tutta la rete sarà a conoscenza della topologia e sarà in grado di adattarsi nel trovare percorsi alternativi in caso di guasti, così da garantire l'affidabilità nel raggiungere i vari nodi nella rete. Prendiamo sempre R8 come esempio di configurazione [figura 6].

<R8>sys Enter system view, return user view with Ctrl+Z. [R8]ospf 1 router-id 8.8.8.8 [R8-ospf-1]area 0 [R8-ospf-1-area-0.0.0.0]network 192.168.83.0 0.0.0.3 [R8-ospf-1-area-0.0.0.0]network 192.168.18.0 0.0.0.255

Figura 6. Configurazione di OSPF su R8.

Dalla figura 6 si può vedere la creazione del processo OSPF con *process ID 1* e *router-id* 8.8.8.8. Il router-id è un identificativo di 32 bit, che ha un ruolo quando sono presenti le elezioni per il DR (quindi non nel nostro caso). Per facilitare la memorizzazione e la lettura nei comandi *display*, si è deciso di usare il valore decimale corrispondente al router nei quattro ottetti. Perciò R8 avrà router-id 8.8.8.8, R3 avrà 3.3.3.3 e così via. Tutti router nella rete dovranno far parte dello stesso processo e della stessa area per avere lo stesso LSDB, cioè la conoscenza dell'intera topologia. Il traffico OSPF per raggiungere la convergenza non incide in maniera significativa sulle prestazioni della rete. Ritornando alla figura, possiamo vedere l'abilitazione delle interfacce che hanno un match con 192.168.83.0 e 192.168.18.0 da parte di R8 tramite il comando *network* nella view del processo OSPF. A destra dell'indirizzo abbiamo la *Wildcard Mask* (32 bit), i cui bit a 0 sono dove deve esserci un match nel confronto tra indirizzi IP, mentre dove sono i bit a 1 non è necessario un match. Nel nostro caso, è semplicemente l'inverso della *Subnet Mask*, perciò dove ho un indirizzo /30 avrò la WCM 0.0.255 (gli otto bit meno significativi a 1).

Una volta configurato ogni router in questo modo, possiamo usare i vari comandi di display per visualizzare le varie informazioni e parametri e per vedere se il tutto funziona ed è stato configurato correttamente. Partiamo con il comando *display ospf brief*, che ci permette varie informazioni generali del processo OSPF appena impostato [figura 7].

```
<R8>dis ospf brief
      OSPF Process 1 with Router ID 8.8.8.8
             OSPF Protocol Information
RouterID: 8.8.8.8
                             Border Router:
Multi-VPN-Instance is not enabled
Global DS-TE Mode: Non-Standard IETF Mode
Graceful-restart capability: disabled
Helper support capability : not configured
Applications Supported: MPLS Traffic-Engineering
   f-schedule-interval: max 10000ms, start 500ms, hold 1000ms
Default ASE parameters: Metric: 1 Tag: 1 Type: 2
Route Preference: 10
ASE Route Preference: 150
SPF Computation Count: 6
RFC 1583 Compatible
Retransmission limitation is disabled
Area Count: 1 Nssa Area Count: 0
ExChange/Loading Neighbors: 0
Process total up interface count: 2
Process valid up interface count: 2
Area: 0.0.0.0
                         (MPLS TE not enabled)
Authtype: None
                  Area flag: Normal
SPF scheduled Count: 6
ExChange/Loading Neighbors: 0
Router ID conflict state: Normal
Area interface up count: 2
Interface: 192.168.83.1 (GigabitEthernet0/0/0) --> 192.168.83.2
Cost: 1 State: P-2-P Typ
Timers: Hello 10 , Dead 40 , Poll
                                                 MTU: 1500
                               Type: P2P
                                    120 , Retransmit 5 , Transmit Delay 1
Interface: 192.168.18.2 (GigabitEthernet0/0/1)
                                 Type: P2P
                                                  MTU: 1500
Cost: 1
               State: P-2-P
Timers: Hello 10 , Dead 40 , Poll
                                    120 , Retransmit 5 , Transmit Delay 1
```

Figura 7. Visualizzazione dei parametri della configurazione di OSPF su R8.

Questo comando ci fornisce tante informazioni, cerchiamo di evidenziare quelle più rilevanti: il process-id, il router-id, il valore di preferenza della rotta OSPF (per default è 10), l'area OSPF e le interfacce del router che ne partecipano, specificando il costo del collegamento, il tipo (P2P poiché lo abbiamo impostato noi tramite il comando *ospf network-type p2p* descritto all'inizio del paragrafo), l'MTU (Maximum Transmission Unit) e i valori dei vari timers utili per mantenere sempre aggiornata la visione della topologia. Ci permette, quindi, di avere una buona visione dell'intera configurazione. Un altro comando utile è *display ospf peer*, che ci permette di visualizzare i vicini del router che partecipano ai suoi stessi processi OSPF [figura 8].

```
<R8>dis ospf peer
      OSPF Process 1 with Router ID 8.8.8.8
            Neighbors
Area 0.0.0.0 interface 192.168.83.1 (GigabitEthernet0/0/0)'s neighbors
Router ID: 3.3.3.3
                             Address: 192.168.83.2
                           Slave
  State: Full Mode:Nbr is
                                    Priority: 1
             BDR: None
                          MTU: 0
    R: None
     d timer due in 40
                         sec
  Retrans timer interval: 5
  Neighbor is up for 00:16:30
  Authentication Sequence: [ 0 ]
```

Figura 8. Visualizzazione dei peer vicini a R8 che fanno parte dello stesso processo OSPF.

Gli attributi più importanti presenti in figura 8 sono: l'area in cui è stabilita la relazione tra i peer, il router-id e l'indirizzo IP del peer (in questo caso è del router R3), lo stato di associazione (se è Full significa che sta funzionando correttamente), l'associazione master/slave per negoziare l'adiacenza e raggiungere lo stato di *Full Adjacency* e anche le assegnazioni dei ruoli DR e BDR (Backup DR) che in questo caso non ci sono perché abbiamo impostato il collegamento come *point-to-point*.

L'ultimo comando che vediamo per la verifica di OSPF è *display ospf lsdb*, che mostra i partecipanti del processo OSPF. Infatti, vediamo dalla figura 9 che sono presenti tutti i router presenti nella topologia.

<r8>dis ospf lsdb</r8>										
os	PF Process 1 wit Link State D	3.8								
	Area	: 0.0.0.0								
Туре	LinkState ID	AdvRouter	Age	Len	Sequence	Metric				
Router	4.4.4.4	4.4.4.4	1064	144	8000000A	1				
Router	2.2.2.2	2.2.2.2	1726	96	80000006	1				
Router	9.9.9.9	9.9.9.9	1793	60	80000003	1				
Router	1.1.1.1	1.1.1.1	1063	96	80000006	1				
Router	8.8.8.8	8.8.8.8	1062	60	80000003	1				
Router	5.5.5.5	5.5.5.5	1726	120	80000008	1				
Router	3.3.3.3	3.3.3.3	1060	120	80000008	1				
Router	10.10.10.10	10.10.10.10	1062	96	80000006	1				

Figura 9. Visualizzazione del LSDB su R8.

Come ulteriore verifica, eseguiamo un *ping* da R8 per testare la raggiungibilità di R9 ed effettuiamo anche l'analisi del percorso del pacchetto tramite il comando *tracert* [figura 10].

```
R8>ping 192.168.59.2
 PING 192.168.59.2: 56 data bytes, press CTRL_C to break
   Reply from 192.168.59.2: bytes=56 Sequence=1 tt1=252 time=60
   Reply from 192.168.59.2: bytes=56 Sequence=2 ttl=252
                                                         time=50
   Reply from 192.168.59.2: bytes=56 Sequence=3 tt1=252
                                                         time
   Reply from 192.168.59.2: bytes=56 Sequence=4 tt1=252 time=90
                                                                 ms
         from 192.168.59.2: bytes=56 Sequence=5 ttl=252 time=70 ms
   -- 192.168.59.2 ping statistics ---
   5 packet(s) transmitted
   5 packet(s) received
   0.00% packet loss
   round-trip min/avg/max = 50/66/90 ms
(R8>tracert 192.168.59.2)
traceroute to 192.168.59.2(192.168.59.2), max hops: 30 ,packet length: 40,pres
s CTRL C to break
1 192.168.83.2 80 ms 50 ms
                             30 ms
2 192.168.34.2 110 ms 192.168.103.2 70 ms 192.168.34.2 50 ms
3 192.168.105.2 80 ms 192.168.45.2 70 ms 192.168.105.2 50 ms
4 192.168.59.2 80 ms 60 ms 70 ms
```

Figura 10. Risultati del ping e del traceroute da R8 a R9, che mostra più percorsi possibili.

Dalla figura 10 si può dedurre che R9 è raggiungibile. È interessante analizzare il risultato del traceroute: il pacchetto, che parte da R8 per raggiungere R9, passa per R3 dal quale si può diramare sui tre link che offre il router, poiché hanno tutti la stessa preferenza e costo, cioè si ha un ECMP (Equal Cost MultiPath). Concentrandoci sul percorso verso R10, vediamo che poi passerà per R5 per arrivare, infine, a R9.

Infine, vediamo l'effetto di OSPF sulla tabella di routing, in particolare su R3 [figure 11 e 12].

<r3>dis ip routing-table Route Flags: R - relay, D - download to fib</r3>										
Routing Tables: Public										
Destinatio	ns : 26		Routes : 3	32						
Destination/Mask	Proto	Pre	Cost	Flags	NextHop	Interface				
127.0.0.0/8	Direct	0	0	D	127.0.0.1	InLoopBack0				
127.0.0.1/32	Direct	0	0	D	127.0.0.1	InLoopBack0				
127.255.255.255/32	Direct	0	0	D	127.0.0.1	InLoopBack0				
192.168.12.0/30	OSPF	10	2	D	192.168.31.2	GigabitEthernet				
4/0/0										
192.168.14.0/30 4/0/0	OSPF	10	2	D	192.168.31.2	GigabitEthernet				
	OSPF	10	2	D	192.168.34.2	GigabitEthernet				
0/0/1										
192.168.18.0/24	OSPF	10	2	D	192.168.83.1	GigabitEthernet				
0/0/0										
192.168.25.0/30	OSPF	10	3	D	192.168.31.2	GigabitEthernet				
17070	OSPF	10	3	D	192.168.103.2	GigabitEthernet				
4/0/1										
	OSPF	10	3	D	192.168.34.2	GigabitEthernet				
0/0/1										
192.168.31.0/30	Direct	0	0	D	192.168.31.1	GigabitEthernet				
4/0/0										
192.168.31.1/32	Direct	0	0	D	127.0.0.1	GigabitEthernet				
4/0/0										
192.168.31.3/32	Direct	0	0	D	127.0.0.1	GigabitEthernet				
4/0/0										
192.168.34.0/30	Direct	0	0	D	192.168.34.1	GigabitEthernet				
0/0/1										
192.168.34.1/32	Direct	0	0	D	127.0.0.1	GigabitEthernet				
0/0/1										
192.168.34.3/32	Direct	0	0	D	127.0.0.1	GigabitEthernet				
0/0/1										
192.168.42.0/30	OSPF	10	2	D	192.168.34.2	GigabitEthernet				
0/0/1										
192,168,45,0/30	OSPF	10	2	D	192,168,34,2	GigabitEthernet				
0/0/1										

Figura 11. Prima parte della tabella di routing di R3.

192.168.59.0/30	OSPF	10	3	D	192.168.103.2	GigabitEthernet
4/0/1						
	OSPF	10	3	D	192.168.34.2	GigabitEthernet
0/0/1						
192.168.83.0/30	Direct	0	0	D	192.168.83.2	GigabitEthernet
0/0/0						
192.168.83.2/32	Direct	0	0	D	127.0.0.1	GigabitEthernet
0/0/0						
192.168.83.3/32	Direct	0	0	D	127.0.0.1	GigabitEthernet
0/0/0						
192.168.92.0/24	OSPF	10	4	D	192.168.103.2	GigabitEthernet
4/0/1						
	OSPF	10	4	D	192.168.34.2	GigabitEthernet
0/0/1						
192.168.103.0/30	Direct	0	0	D	192.168.103.1	GigabitEthernet
4/0/1						
192.168.103.1/32	Direct	0	0	D	127.0.0.1	GigabitEthernet
4/0/1						
192.168.103.3/32	Direct	0	0	D	127.0.0.1	GigabitEthernet
4/0/1						
192.168.104.0/30	OSPF	10	2	D	192.168.103.2	GigabitEthernet
4/0/1						
	OSPF	10	2	D	192.168.34.2	GigabitEthernet
0/0/1						
192.168.105.0/30	OSPF	10	2	D	192.168.103.2	GigabitEthernet
4/0/1						
255.255.255.255/32	Direct	0	0	D	127.0.0.1	InLoopBack0

Figura 12. Seconda parte della tabella di routing di R3.

È immediato vedere quali rotte sono state apprese attraverso OSPF. Inoltre, si può notare ancora la presenza di ridondanza, poiché per la stessa destinazione ho più *NextHop* possibili (con stesso valore di preferenza e stesso costo).

Come ultima cosa, vogliamo testare la capacità di adattamento alle modifiche nella rete. Per farlo, disattiviamo l'interfaccia GigabitEthernet 0/0/1 di R3, che collega R3 a R4, tramite il comando *shutdown* nella view di interfaccia e cerchiamo di raggiungere comunque R4. Abbiamo utilizzato il comando *tracert* per raggiungere 192.168.45.1 per verificare la raggiungibilità [figura 13].

Figura 13. Traceroute da R3 a R4 dopo lo shutdown dell'interfaccia verso R4.

7.3. Configurazione DHCP nelle reti Customer

Per l'assegnazione automatica degli indirizzi IP agli host nelle reti customer, ci affidiamo a DHCP (da implementare nei router CE R8 e R9). Si è deciso di implementare una *DHCP Global Pool Configuration*, piuttosto che quella di interfaccia. Questa scelta è stata fatta per rendere più comoda e immediata l'aggiunta di ulteriori host nelle reti Customer, poiché non serve rimettere mano alla configurazione. Questo perché la configurazione globale consente di ottenere indirizzi IP (non limitati dallo spazio degli indirizzi dell'interfaccia, diversamente da quanto accade con Interface Pool Configurazione di interfaccia consente di assegnare indirizzi IP solo ai terminali collegati allo stesso segmento di rete e ciò porta a dover rivedere alla configurazione se si vogliono aggiungere altri host collegati su segmenti diversi.

Nella figura 14 mostriamo il processo di configurazione di DHCP nel router R8.



Figura 14. Configurazione completa di DHCP su R8 per l'assegnazione automatica degli indirizzi IP della rete Customer.

Innanzitutto, bisogna abilitare DHCP sul router tramite il comando *dhcp enable* nella view di sistema, poi si crea il pool di indirizzi con *ip pool <nome_pool*>. Una volta dentro la configurazione del pool appena creato, si imposta il range di indirizzi IP (specificando la maschera di sottorete), l'indirizzo IP del gateway (in questo caso è l'indirizzi di GigabitEthernet 0/0/1 di R8) e il periodo di rilascio degli indirizzi assegnati (in questo caso si è scelto di 1 giorno per comodità nel testare la rete su eNSP). Infine, bisogna andare nella view di interfaccia nella quale si vuole abilitare DHCP e si usa il comando *dhcp select global* per selezionare la Global Pool Interface. R9 si configura in modo analogo, ciò che cambierà sarà il range (192.168.92.0 mask 24) e l'indirizzo del gateway (192.168.92.1). Una volta fatto tutto questo, gli host saranno in grado di acquisire un indirizzo IP dal pool e, quindi, di collegarsi all'intera rete.

Per verificare la configurazione di DHCP, invochiamo il comando *display ip pool* [figura 15] e, come ultima cosa, verifichiamo sul PC (che deve essere impostato per usara DHCP [figura 16]) se ha acquisito automaticamente un indirizzo IP tramite il comando *ipconfig* [figura 17].

<r8>dis ip pool</r8>				
Pool-name	: pool1			
Pool-No	: 0			
Position	: Local	Sta	tus	: Unlocked
Gateway-0	: 192.16	58.18.2		
Mask	: 255.25	5.255.0		
VPN instance	:			
IP address St	atistic			
Total	:253			
Used	:1	Idle	:252	
Expired	:0	Conflict	:0	Disable :0

Figura 15. Visualizzazione del pool configurato su R8.

🗧 PC1			
Basic Config Com	mand MCPacket UdpP	acket Console	
Host Name:	I		
MAC Address:	54-89-98-93-43-C6		
IPv4 Configuratio	n		
◯ Static	DHCP	Obtain DNS server addre	ess automatically
IP Address:		DNS1: .	
Subnet Mask:		DNS2:	
Gateway:			

Figura 16. Finestra di impostazione di PC1 dove è configurata l'impostazione DHCP.

PC>ipconfig	
Link local IPv6 address	fe80::5689:98ff:fe93:43c6
IPv6 address	:: / 128
IPv6 gateway	::
IPv4 address	192.168.18.254
Subnet mask	255.255.255.0
Gateway Physical address DNS server	192.168.18.2 54-89-98-93-43-06

Figura 17. Informazioni date dal comando ipconfig su PC1.

Dalla figura 17 possiamo vedere che l'host ha correttamente impostato il gateway 192.168.18.2 (interfaccia interna di R8) ed ha acquisito dal pool l'indirizzo IP 192.168.18.254.

Ora verifichiamo il funzionamento della rete facendo un ping da PC1 per raggiungere PC2, così che i pacchetti debbano attraversare l'intera topologia per raggiungere la destinazione. Prima di eseguire il ping, dobbiamo apprendere l'indirizzo IP di PC2, anch'esso acquisito in maniera automatica con R9 che funge da DHCP server. Mostriamo il risultato del ping nella figura 18.

```
PC>ping 192.168.92.254
Ping 192.168.92.254: 32 data bytes, Press Ctrl_C to break
Request timeout!
From 192.168.92.254: bytes=32 seq=2 ttl=123 time=63 ms
From 192.168.92.254: bytes=32 seq=3 ttl=123 time=62 ms
From 192.168.92.254: bytes=32 seq=4 ttl=123 time=47 ms
From 192.168.92.254: bytes=32 seq=5 ttl=123 time=63 ms
--- 192.168.92.254 ping statistics ---
5 packet(s) transmitted
4 packet(s) received
20.00% packet loss
round-trip min/avg/max = 0/58/63 ms
```

Figura 18. Risultati del ping da PC1 a PC2.

Vediamo che il ping verso PC2 ha avuto successo. Però, il primo tentativo non ha avuto successo, dandoci come messaggio di errore *Request timeout*. Questo indica che si è superato il tempo limite per ricevere una Echo Reply da PC2. Per capire il motivo di questo evento, sfrutteremo Wireshark sulle interfacce dei due PC per vedere in dettaglio il flusso di traffico [figura 19]. Partiamo dall'interfaccia da cui parte il ping, cioè PC1 (Ethernet 0/0/1).

	not ospf										
N).	Time	Source	Destination	Protocol	Length Info					
	2:	83.125000	HuaweiTe_93:43:c6	Broadcast	ARP	60 Who has 192.168.18.2? Tell 192.168.18.254					
	22	83.203000	HuaweiTe_73:76:06	HuaweiTe_93:43:c6	ARP	60 192.168.18.2 is at 00:e0:fc:73:76:06					
Г	- 23	83.203000	192.168.18.254	192.168.92.254	ICMP	74 Echo (ping) request id=0x3474, seq=1/256, ttl=128 (no response found)				
	24	85.203000	192.168.18.254	192.168.92.254	ICMP	74 Echo (ping) request id=0x3674, seq=2/512, ttl=128 (reply in 25)					
	25	85.390000	192.168.92.254	192.168.18.254	ICMP	74 Echo (ping) reply id=0x3674, seq=2/512, ttl=123 (request in 24)					
	26	5 86.390000	192.168.18.254	192.168.92.254	ICMP	74 Echo (ping) request id=0x3774, seq=3/768, ttl=128 (reply in 27)					
	27	86.515000	192.168.92.254	192.168.18.254	ICMP	74 Echo (ping) reply id=0x3774, seq=3/768, ttl=123 (request in 26)					
	28	87.531000	192.168.18.254	192.168.92.254	ICMP	74 Echo (ping) request id=0x3874, seq=4/1024, ttl=128 (reply in 29)					
	29	87.594000	192.168.92.254	192.168.18.254	ICMP	74 Echo (ping) reply id=0x3874, seq=4/1024, ttl=123 (request in 28)					
	30	88.609000	192.168.18.254	192.168.92.254	ICMP	74 Echo (ping) request id=0x3a74, seq=5/1280, ttl=128 (reply in 31)					
L	- 31	88.890000	192.168.92.254	192.168.18.254	ICMP	74 Echo (ping) reply id=0x3a74, seq=5/1280, ttl=123 (request in 30)					
				74 hotes and (50	0.1.2.4		_				
1	Fra	me 23: 74 t	bytes on wire (592 bits),	74 bytes captured (59	2 D1ts)	on interface -, 10 0					
>	Eth	ernet II, S	Src: HuaweiTe_93:43:c6 (54	:89:98:93:43:c6), Dst	: Huawei	tiTe_73:76:06 (00:e0:fc:73:76:06)					
>	Int	ernet Proto	ocol Version 4, Src: 192.1	68.18.254, Dst: 192.1	68.92.25	54					
>	Int	Internet Control Message Protocol									

Figura 19. Cattura del traffico sull'interfaccia Ethernet 0/0/1 di PC1 generato dall'operazione di ping tra PC1 e PC2.

Abbiamo filtrato il traffico OSPF, così da non visualizzare i vari *Hello Packet*, che in questo momento non ci interessa. Possiamo notare che, prima di effetuare l'Echo Request, PC1 cerca di apprendere tramite il protocollo ARP l'indirizzo MAC del suo gateway di uscita, di cui

conosce l'indirizzo IP (192.168.18.2). Questo processo avviene solo la prima volta che si genera traffico da PC1, poiché deve riempire la *MAC Address Table* per riuscire a mandare il frame al gateway di uscita. Possiamo infatti vedere che l'indirizzo MAC di destinazione del frame che trasporta l'Echo Request è quello del router R8, appreso grazie alla risposta del router *all'ARP Request* generata da PC1. Anche da qui possiamo vedere che il primo tentativo di ping non ha avuto una risposta. Da questa cattura non possiamo ancora capirne il motivo, passiamo quindi a PC2 [figura 20].

Γ.	not ospf									
No		Time	Source	Destination	Protocol	Length	Info			
	7	47.312000	HuaweiTe_d2:4d:b8	Broadcast	ARP	60	Who has 192.168.92.254? Tell 192.168.92.1			
	8	47.312000	HuaweiTe_89:65:34	HuaweiTe_d2:4d:b8	ARP	60	192.168.92.254 is at 54:89:98:89:65:34			
*	9	48.641000	192.168.18.254	192.168.92.254	ICMP	74	Echo (ping) request id=0x3674, seq=2/512, ttl=123 (reply in 10)			
4	10	48.641000	192.168.92.254	192.168.18.254	ICMP	74	Echo (ping) reply id=0x3674, seq=2/512, ttl=128 (request in 9)			
	11	49.781000	192.168.18.254	192.168.92.254	ICMP	74	Echo (ping) request id=0x3774, seq=3/768, ttl=123 (reply in 12)			
	12	49.781000	192.168.92.254	192.168.18.254	ICMP	74	Echo (ping) reply id=0x3774, seq=3/768, ttl=128 (request in 11)			
	13	50.859000	192.168.18.254	192.168.92.254	ICMP	74	Echo (ping) request id=0x3874, seq=4/1024, ttl=123 (reply in 14)			
	14	50.859000	192.168.92.254	192.168.18.254	ICMP	74	Echo (ping) reply id=0x3874, seq=4/1024, ttl=128 (request in 13)			
	15	52.156000	192.168.18.254	192.168.92.254	ICMP	74	Echo (ping) request id=0x3a74, seq=5/1280, ttl=123 (reply in 16)			
	16	52.156000	192.168.92.254	192.168.18.254	ICMP	74	Echo (ping) reply id=0x3a74, seq=5/1280, ttl=128 (request in 15)			
			the second of (500 bits) 7	A human analysis of (500	Laters -		f			
1	Fra	ne 9: 74 by	/tes on wire (592 bits), /	4 bytes captured (592	DITS) O	n inter	Tace -, 10 0			
>	Eth	ernet II, S	Src: HuaweiTe_d2:4d:b8 (00	:e0:fc:d2:4d:b8), Dst	: Huawei	Te_89:6	5:34 (54:89:98:89:65:34)			
>	Int	ernet Proto	ocol Version 4, Src: 192.1	68.18.254, Dst: 192.1	68.92.25	4				
>	Int	ernet Contr	rol Message Protocol							

Figura 20. Cattura del traffico sull'interfaccia Ethernet 0/0/1 di PC2 generato dall'operazione di ping tra PC1 e PC2.

Filtriamo anche qui il traffico OSPF. Notiamo che R9 (192.168.92.1) deve eseguire un ARP Request per scoprire il MAC address di PC2. Quindi, il primo tentativo di ping non ottiene risposta da PC2, perché R9 si vede arrivare l'Echo Request da PC1, ma in quel momento non riesce a inoltrarlo subito perché deve prima conoscere il MAC address di PC2 per inviare la richiesta di ping attraverso un frame. Infatti, una volta appreso, PC2 riceve l'Echo Request e riesce a mandare l'Echo Reply con successo. Per i ping successivi tra PC1 e PC2, non ci sarà nessun Request Timeout poiché non sarà necessario eseguire il processo ARP, siccome la MAC Address Table è stata riempita con gli indirizzi fisici necessari per la comunicazione.

7.4. Configurazione di percorsi predefiniti

Come richiesto dalla traccia, la comunicazione tra le due reti Customer (formate da PC1, R8 e PC2, R9) deve avvenire sul percorso R3-R1-R2-R5 (percorso principale) oppure, in caso di un fail del primo percorso, su R3-R10-R5 (percorso di backup). Per fare questo, si è pensato di impostare delle rotte statiche nei router chiamati in causa e modificarne il valore di preferenza (ricordiamo che il valore di preferenza più bassa indica una priorità più alta). Sappiamo che il valore minimo di preferenza presente nelle routing-table è 10 (a parte i link diretti che hanno un valore pari a 0), che è il valore di default delle rotte apprese da OSPF. Per questo motivo, useremo il valore di preferenza 8 per il percorso principale, mentre useremo 9 per quello di

backup. Abbiamo deciso di non usare valori più bassi per lasciar spazio a implementazioni future di percorsi con maggiore priorità.

Tramite il comando *ip route-static*, andremo a configurare le rotte sui router R3, R1, R2, R5, R10 con le giuste preferenze. Partiamo con l'esempio di R3 [figura 21], che è un nodo in cui avviene la biforcazione del percorso principale e quello di backup.

```
[R3]ip route-static 192.168.59.0 30 192.168.31.2 preference 8
[R3]ip route-static 192.168.92.0 24 192.168.31.2 preference 8
[R3]ip route-static 192.168.92.0 24 192.168.103.2 preference 9
[R3]ip route-static 192.168.59.0 30 192.168.103.2 preference 9
```

Figura 21. Configurazione delle rotte statiche (principale e di backup) su R3 verso la rete Customer 2.

Le prime due righe di comando ci dicono che per raggiungere la rete Customer 2 (192.168.59.0/30 e 192.168.92.0/24) i pacchetti devono avere R1 come NextHop (192.168.31.2), altrimenti il percorso alternativo è dato da R10 (192.168.103.2). Su R3 non c'è bisogno di configurare la rotta verso la rete Customer 1, poiché ha R8 direttamente collegato ad esso.

Guardiamo ora R1 come caso di configurazione di un nodo intermedio al percorso [figura 21].



Figura 22. Configurazione delle rotte statiche (principale e di backup) su R1 verso le reti Customer 1 e 2.

Su R1 ci deve passare solo il percorso principale, infatti tutte le rotte hanno come valore preferenza 8. Le prime due righe di comando specificano la direzione per raggiungere la rete Customer 2, cioè verso R2 (192.168.12.2). Le restanti due righe, invece, ci indicano R3 (192.168.31.1) come NextHop per raggiungere la rete Customer 1.

Come prima verifica, possiamo eseguire il comando *display ip routing-table*, ad esempio su R3 [figura 23].

192.168.42.0/30	OSPF	10	2	D	192.168.34.2	GigabitEthernet
192.168.45.0/30	OSPF	10	2	D	192.168.34.2	GigabitEthernet
0/0/1 192.168.59.0/30	Static	8	0	RD	192.168.31.2	GigabitEthernet
4/0/0 192.168.83.0/30	Direct	0	0	D	192.168.83.2	GigabitEthernet
0/0/0 192.168.83.2/32	Direct	0	0	D	127.0.0.1	GigabitEthernet
0/0/0	Direct	0	0	р	127.0.0.1	GigabitEthernet
0/0/0	Statia		0		102,168,21,2	CirchitEtherest
4/0/0	Static	•	0	RD	192.100.31.2	GigabitEthernet

Figura 23. Tabella di routing di R3 per visualizzare la presenza delle rotte statiche configurate.

Nella tabella di routing è presente soltanto il percorso principale verso la rete Customer 2, poiché ci mostra solo i percorsi con maggiore priorità in questo momento, che in questo caso è il percorso principale con valore di preferenza 8.

Come ultime verifiche, facciamo un traceroute verso R9 (192.168.59.2) da R3, per vedere se i pacchetti utilizzano effettivamente il percorso principale impostato [figura 24].

<r3>tracert 192</r3>	.168.59.3	2				
<pre>traceroute to s CTRL_C to brea</pre>	192.168 ak	.59.2(1	92.168.59.2),	max hops: 30	,packet length:	40,pres
1 192.168.31.2	230 ms	60 ms	30 ms			
2 192.168.12.2	100 ms	40 ms	50 ms			
3 192.168.25.2	120 ms	50 ms	50 ms			
4 192.168.59.2	190 ms	90 ms	50 ms			

Figura 24. Traceroute per la verifica della rotta principale partendo da R3.

Vediamo dalla figura 24 che il pacchetto passa attraverso i router R1, R2, R5 e arriva a R9, proprio come voluto dalle specifiche di progetto. Ora, spegniamo il collegamento tra R3 e R1 per testare anche il percorso di backup [figura 25].

```
<R3>tracert 192.168.59.2

traceroute to 192.168.59.2(192.168.59.2), max hops: 30 ,packet length: 40,pres

s CTRL_C to break

1 192.168.103.2 130 ms 60 ms 50 ms

2 192.168.105.2 110 ms 60 ms 40 ms

3 192.168.59.2 70 ms 60 ms 60 ms
```

Figura 25. Traceroute per la verifica della rotta di backup partendo da R3 dopo lo shutdown dell'interfaccia GE 4/0/0.

Il pacchetto passa per R10, R5 e arriva a R9, confermando la corretta configurazione.

È interessante fare una valutazione *real-time* per analizzare la reazione della rete a questo cambiamento di topologia. Eseguiamo un'operazione di ping da PC1 a PC2 con il comando ping 192.168.92.254 -c 1000, così da inviare 1000 Echo Request. Abbiamo impostato un numero così elevato di Echo Request per avere abbastanza tempo per analizzare il comportamento prima e dopo lo shutdown dell'interfaccia. Nella figura 26 vediamo i risultati dell'operazione di ping descritta prima e l'operazione di traceroute.

🔁 PC1							
De la Cart	0	MOD			0		
Basic Config	Command	WCPacket	UdpP	acket	Cons	ole	
From 192.1	68.92.254:	bytes=32	seq=37	ttl=1	.22 tim	ie=63	ms
Request tin	meout!						
Request tin	meout!						
Request tin	meout!					1.04	
From 192.1	68.92.254:	bytes=32	seq=41	ttl=1	21 tim	le=109	9 ms
From 192.1	68.92.254:	bytes=32	seq=42		21 tim	le=62	ms
From 192.1	68.92.254:	bytes=32	seq=43		21 tim	te=79	ms
From 192.1	68.92.254:	bytes=32	seq=44		21 tim	le=62	ms
From 192.1	68.92.254:	bytes=32	seq=45		21 tim	le=94	ms
From 192.1	60.92.254:	bytes=32	seq-40	++1-1	21 tim	1e - 70	ms 0 mo
From 192.1	60.92.254:	bytes=32	seq-47	++1=1	21 tim	e-10:	9 ms
From 192.1	60.92.251.	bytes=32	seq=10	++1=1	21 tim	0=62	me
From 192.1	68 92 254	bytes=32	seg=19	++1=1	21 tim	r = 102	9 mg
From 192.1	68.92.254	bytes=32	seg=51	ttl=1	21 tim	e=63	ms
From 192.1	68.92.254:	bytes=32	seg=52	tt1=1	21 tim	1e=94	ms
From 192.1	68.92.254:	bytes=32	seg=53	tt]=1	21 tim	e=62	ms
From 192.1	68.92.254:	bytes=32	seg=54	ttl=1	21 tim	ie=63	ms
From 192.1	68.92.254:	bvtes=32	sea=55	ttl=1	21 tim	e=109	9 ms
From 192.1	68.92.254:	bytes=32	seq=56	ttl=1	.21 tim	te=93	ms
From 192.1	68.92.254:	bytes=32	seq=57	ttl=1	.21 tim	ue=11(0 ms
192.16	8.92.254 p:	ing statis	stics				
57 packet	t(s) trans	nitted					
53 packer	t(s) receiv	ved					
7.02% pa	cket loss						
round-tr:	ip min/avg,	/max = 62/	/84/203	ms			
PC>tracert	192.168.93	2.254					
traceroute	to 192.16	8.92.254,	8 hops	max			
(ICMP), pr	ess Ctrl+C	to stop					
1 192.16	8.18.2 1	5 ms 32 n	ns <1 r	ns			
2 192.16	8.83.2 3	1 ms 31 n	ns 63 r	ns			
3 192.16	8.103.2	62 ms 47	ms 47	ms			
4 192.16	8.105.2	94 ms 78	ms 62	ms			
5 192.16	8.59.2 1	10 ms 78	ms 109	9 ms			
6 192.16	8.92.254	110 ms (oz ms	od ms			

Figura 26. Ping tra PC1 e PC2 durante l'interruzione del collegamento del percorso principale e traceroute dopo il cambiamento della topologia.

Possiamo capire il momento in cui è avvenuta l'interruzione del percorso principale vedendo quando riceviamo il messaggio di errore Request Timeout. Dopo aver perso 3 pacchetti, la rete riesce a adattarsi al cambio di topologia, scegliendo il percorso di backup come dimostrato dal risultato traceroute nella figura 26. Possiamo visualizzare meglio questo evento tramite delle catture con Wireshark.

U	not c	ospf											
N	o.	Time	Source	Destination	Protocol	Length	Info						
	75	75.594000	192.168.18.254	192.168.92.254	ICMP	74	Echo	(ping)	request	id=0xcddd,	seq=34/8704,	ttl=128	(reply in 76)
	76	75.656000	192.168.92.254	192.168.18.254	ICMP	74	Echo	(ping)	reply	id=0xcddd,	seq=34/8704,	ttl=122	(request in 75)
	77	76.656000	192.168.18.254	192.168.92.254	ICMP	74	Echo	(ping)	request	id=0xcedd,	seq=35/8960,	ttl=128	(reply in 78)
	78	76.734000	192.168.92.254	192.168.18.254	ICMP	74	Echo	(ping)	reply	id=0xcedd,	seq=35/8960,	ttl=122	(request in 77)
	79	77.750000	192.168.18.254	192.168.92.254	ICMP	74	Echo	(ping)	request	id=0xd0dd,	seq=36/9216,	ttl=128	(reply in 80)
	80	77.812000	192.168.92.254	192.168.18.254	ICMP	74	Echo	(ping)	reply	id=0xd0dd,	seq=36/9216,	ttl=122	(request in 79)
	81	78.828000	192.168.18.254	192.168.92.254	ICMP	74	Echo	(ping)	request	id=0xd1dd,	seq=37/9472,	ttl=128	(reply in 82)
	82	78.875000	192.168.92.254	192.168.18.254	ICMP	74	Echo	(ping)	reply	id=0xd1dd,	seq=37/9472,	ttl=122	(request in 81)
	83	79.891000	192.168.18.254	192.168.92.254	ICMP	74	Echo	(ping)	request	id=0xd2dd,	seq=38/9728,	ttl=128	(no response found!)
	84	81.891000	192.168.18.254	192.168.92.254	ICMP	74	Echo	(ping)	request	id=0xd4dd,	seq=39/9984,	ttl=128	(no response found!)
	86	83.891000	192.168.18.254	192.168.92.254	ICMP	74	Echo	(ping)	request	id=0xd6dd,	seq=40/10240,	, ttl=128	(no response found!
	87	85.906000	192.168.18.254	192.168.92.254	ICMP	74	Echo	(ping)	request	id=0xd8dd,	seq=41/10496,	, ttl=128	(reply in 88)
	88	86.000000	192.168.92.254	192.168.18.254	ICMP	74	Echo	(ping)	reply	id=0xd8dd,	seq=41/10496,	, ttl=121	(request in 87)
	89	87.000000	192.168.18.254	192.168.92.254	ICMP	74	Echo	(ping)	request	id=0xd9dd,	seq=42/10752,	, ttl=128	(reply in 90)
	90	87.062000	192.168.92.254	192.168.18.254	ICMP	74	Echo	(ping)	reply	id=0xd9dd,	seq=42/10752,	, ttl=121	(request in 89)
	91	88.062000	192.168.18.254	192.168.92.254	ICMP	74	Echo	(ping)	request	id=0xdadd,	seq=43/11008,	, ttl=128	(reply in 92)
	92	88.141000	192.168.92.254	192.168.18.254	ICMP	74	Echo	(ping)	reply	id=0xdadd,	seq=43/11008,	, ttl=121	(request in 91)
	93	89.141000	192.168.18.254	192.168.92.254	ICMP	74	Echo	(ping)	request	id=0xdbdd,	seq=44/11264,	, ttl=128	(reply in 94)
	94	89.203000	192.168.92.254	192.168.18.254	ICMP	74	Echo	(ping)	reply	id=0xdbdd,	seq=44/11264,	, ttl=121	(request in 93)
	95	90.219000	192.168.18.254	192.168.92.254	ICMP	74	Echo	(ping)	request	id=0xdcdd,	seq=45/11520,	, ttl=128	(reply in 96)
	96	90.297000	192.168.92.254	192.168.18.254	ICMP	74	Echo	(ping)	reply	id=0xdcdd,	seq=45/11520,	, ttl=121	(request in 95)
	97	91.312000	192.168.18.254	192.168.92.254	ICMP	74	Echo	(ping)	request	id=0xdddd,	seq=46/11776,	, ttl=128	(reply in 98)
<													
>	Fra	me 83: 74 l	oytes on wire (592 bits),	74 bytes captured (59	2 bits)	on inte	rface	-, id	0				
>	Eth	ernet II, S	orc: HuaweiTe 93:43:c6 (54	:89:98:93:43:c6), Dst	: Huawei	Te 73:7	6:06	(00:e0:	fc:73:76:	06)			
)	Int	ernet Proto	col Version 4, Src: 192.1	.68.18.254, Dst: 192.1	68.92.25	4				,			
		And the first of the second seco											

Figura 27. Cattura del traffico generato dal ping tra PC1 e PC2 durante l'operazione di shutdown del collegamento.

La figura 27 mostra la cattura sull'interfaccia Ethernet 0/0/1 di PC1, confermando quanto detto prima sulla capacità di adattamento della nostra rete. Infine, vogliamo mostrare anche il processo di traceroute in Wireshark dopo che si è passati al percorso di backup.

not ospf					
No. Time	Source	Destination	Protocol	Length Info	
126 116.422	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf6dd, seq=1/256	, ttl=1 (no response found!)
127 116.437	192.168.18.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
128 116.453	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf6dd, seq=2/512	!, ttl=1 (no response found!)
129 116.469	192.168.18.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
130 116.469	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf6dd, seq=3/768	, ttl=1 (no response found!)
131 116.469	192.168.18.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
132 116.484	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf6dd, seq=1/256	, ttl=2 (no response found!)
133 116.500	192.168.83.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
134 116.516	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf6dd, seq=2/512	<pre>!, ttl=2 (no response found!)</pre>
135 116.531	192.168.83.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
136 116.547	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf6dd, seq=3/768	, ttl=2 (no response found!)
137 116.594	192.168.83.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
138 116.594	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf6dd, seq=1/256	, ttl=3 (no response found!)
139 116.656	192.168.103.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
140 116.656	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf6dd, seq=2/512	<pre>!, ttl=3 (no response found!)</pre>
141 116.703	192.168.103.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
142 116.703	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf7dd, seq=3/768	<pre>, ttl=3 (no response found!)</pre>
143 116.750	192.168.103.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
144 116.750	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf7dd, seq=1/256	<pre>, ttl=4 (no response found!)</pre>
145 116.844	192.168.105.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
146 116.844	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf7dd, seq=2/512	<pre>!, ttl=4 (no response found!)</pre>
147 116.922	192.168.105.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
148 116.937	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf7dd, seq=3/768	<pre>, ttl=4 (no response found!)</pre>
149 116.984	192.168.105.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
150 116.984	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf7dd, seq=1/256	, ttl=5 (no response found!)
151 117.094	192.168.59.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
152 117.094	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf7dd, seq=2/512	, ttl=5 (no response found!)
153 117.172	192.168.59.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
154 117.172	192.168.18.254	192.168.92.254	ICMP	106 Echo (ping) request id=0xf7dd, seq=3/768	, ttl=5 (no response found!)
155 117.281	192.168.59.2	192.168.18.254	ICMP	70 Time-to-live exceeded (Time to live excee	ded in transit)
<	100 100 10 004	100 100 00 004	TOND	400 F.L. (.:) id 0	111 C (1111. 11 107)
> Frame 83: 74	bytes on wire (592 bits),	74 bytes captured (59	02 bits)	on interface -, id 0	
> Ethernet II.	Src: HuaweiTe 93:43:c6 (54	4:89:98:93:43:c6). Dst	: Huawei	Te 73:76:06 (00:e0:fc:73:76:06)	
C Thernet (eth) 14 hyte				Pacchetti: 164 : visualizzati: 148 (90.2%)

Figura 28. Cattura dell'operazione di traceroute eseguito da PC1 verso PC2.

Nella figura 28 vediamo che l'operazione di traceroute viene eseguita da PC1 facendo un'operazione di ping verso PC2, però usando un TTL crescente finché il pacchetto non raggiungerà la destinazione. Da questa cattura, però, non si riesce ad apprezzare il cambiamento di percorso. Per questo motivo, mostriamo nella figura 29 la cattura del traffico generato dal traceroute eseguito su R8.

No.	Time	Source	Destination	Protocol	Length	Info
5	0 110.766	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33434 Len=12
5	1 110.797	192.168.83.2	192.168.83.1	ICMP	70) Time-to-live exceeded (Time to live exceeded in transit)
5	2 110.813	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33435 Len=12
5	3 110.828	192.168.83.2	192.168.83.1	ICMP	70) Time-to-live exceeded (Time to live exceeded in transit)
5	4 110.844	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33436 Len=12
5	5 110.859	192.168.83.2	192.168.83.1	ICMP	70) Time-to-live exceeded (Time to live exceeded in transit)
5	6 110.875	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33437 Len=12
5	7 110.938	192.168.103.2	192.168.83.1	ICMP	70) Time-to-live exceeded (Time to live exceeded in transit)
5	8 110.953	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33438 Len=12
5	9 111.000	192.168.103.2	192.168.83.1	ICMP	70) Time-to-live exceeded (Time to live exceeded in transit)
6	0 111.016	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33439 Len=12
6	1 111.031	192.168.103.2	192.168.83.1	ICMP	70) Time-to-live exceeded (Time to live exceeded in transit)
6	2 111.047	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33440 Len=12
6	3 111.109	192.168.105.2	192.168.83.1	ICMP	70) Time-to-live exceeded (Time to live exceeded in transit)
6	4 111.125	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33441 Len=12
6	5 111.172	192.168.105.2	192.168.83.1	ICMP	70) Time-to-live exceeded (Time to live exceeded in transit)
6	6 111.219	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33442 Len=12
6	7 111.344	192.168.105.2	192.168.83.1	ICMP	70) Time-to-live exceeded (Time to live exceeded in transit)
6	8 111.375	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33443 Len=12
6	9 111.453	192.168.59.2	192.168.83.1	ICMP	70) Time-to-live exceeded (Time to live exceeded in transit)
7	0 111.484	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33444 Len=12
7	1 111.547	192.168.59.2	192.168.83.1	ICMP	70) Time-to-live exceeded (Time to live exceeded in transit)
7	2 111.563	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33445 Len=12
7	3 111.609	192.168.59.2	192.168.83.1	ICMP	70) Time-to-live exceeded (Time to live exceeded in transit)
7	4 111.625	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33446 Len=12
7	5 111.703	192.168.92.254	192.168.83.1	ICMP	70) Destination unreachable (Port unreachable)
7	6 111.719	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33447 Len=12
7	7 111.766	192.168.92.254	192.168.83.1	ICMP	70	Destination unreachable (Port unreachable)
7	8 111.781	192.168.83.1	192.168.92.254	UDP	60) 30037 → 33448 Len=12
7	9 111.844	192.168.92.254	192.168.83.1	ICMP	70	Destination unreachable (Port unreachable)

Figura 29. Cattura del traffico sull'interfaccia GE 0/0/0 di R8 durante l'operazione di traceroute verso PC2.

Per eseguire il traceroute verso PC2, R8 invia un pacchetto contente UDP con TTL crescente e con una *destination port* che non esisterà nell'host di destinazione, così che, una volta raggiunto, si riceva su R8 il messaggio di errore *port unreachable*, indicando che si è arrivati all'indirizzo di destinazione. Il risultato del traceroute nella figura 29 è più comprensibile rispetto a quello della figura 28. Infatti, basta guardare l'indirizzo IP di sorgente dei messaggi di errore ICMP per capire il percorso intrapreso dal pacchetto per raggiungere PC2: R3 (192.168.83.2), R10 (192.168.103.2), R5 (192.168.105.2), R9 (192.168.59.2) ed infine PC2 (192.168.92.254). Questo è proprio il percorso di backup che abbiamo configurato nella rete.

7.5. Configurazione del Tunnel GRE

Arrivati a questo punto, ipotizziamo che le reti Customer diventino parte della stessa azienda. Per far comunicare entrambe le reti Customer, precedentemente abbiamo attivato lo stesso processo OSPF in ogni dispositivo della topologia, compresi i router CE che "pubblicizzano" le reti aziendali. Però in questo modo si rivela a ISP tutta la struttura interna dal lato Customer, permettendo a ISP di fare routing nelle reti aziendali. Per rendere privata la topologia interna aziendale, si è scelto di utilizzare il Tunnel GRE tra i router CE R8 e R9. Quindi, dobbiamo creare un'interfaccia virtuale tunnel in entrambi i router e un nuovo processo OSPF in esecuzione sulle interfacce aziendali e l'interfaccia tunnel dei router. Riassumendo, si esegue un processo OSPF all'interno della rete aziendale, indipendente da quello presente nella rete ISP (diversi processi OSPF hanno diversi LSDB). In questo modo, gli update dell'OSPF aziendale passeranno nel Tunnel GRE.

Prima di procedere alla configurazione del secondo processo OSPF, bisogna togliere da R8 e R9 le interfacce relative alle reti aziendali da OSPF 1, che sono rispettivamente 192.168.18.0/24 e 192.168.92.0/24. Per farlo bisogna ritornare nella view di OSPF 1 ed eseguire il comando *undo network* <*IP_network*> <*Wildcard_mask*>, dove si inserisce l'indirizzo di rete aziendale impostato nel paragrafo 7.2. Come esempio, usiamo il router CE R8 [figura 30].



Figura 30. Disabilitazione dal processo OSPF 1 delle interfacce relativa agli indirizzi IP 192.168.18.X di R8.

Una volta tolte disabilitate le interfacce relative alle reti aziendali dal processo OSPF 1, si può passare alla configurazione del Tunnel GRE. Prima di tutto, bisogna creare l'interfaccia virtuale

Tunnel sul router CE e poi impostare i suoi parametri. Prendiamo sempre come esempio R8 [figura 31].



Figura 31. Creazione e configurazione dell'interfaccia tunnel virtuale su R8.

Con la prima riga di comando creiamo l'interfaccia virtuale Tunnel 0/0/1 e ci permette di entrare nella sua view di interfaccia, dove definiamo l'IP address dell'estremo del tunnel, il protocollo di incapsulamento usato dal tunnel (GRE), l'indirizzo IP di sorgente e destinazione che troveremo nell'header IP più esterno del pacchetto (è il contenitore esterno del pacchetto che viaggia sul tunnel).

Ora, configuriamo un secondo processo OSPF per instradare il traffico aziendale nell'interfaccia Tunnel. Come sempre prendiamo come caso R8 [figura 32].



Figura 32. Creazione e configurazione del secondo processo OSPF su R8, abilitando le interfacce tunnel e Customer.

Abbiamo creato il processo OSPF2 e a cui devono far parte le interfacce aziendali dalla parte del Customer 1 e l'interfaccia Tunnel. Notiamo che, in questo caso, abbiamo usato la Wildcard Mask 0.0.0.0, così da selezionare quelle interfacce precise che dovranno partecipare a questo secondo processo creato per implementare il Tunnel GRE tra le due reti Customer. Facendo così, siamo sicuri che soltanto queste due precise interfacce saranno abilitate al processo OSPF 2, anche dopo un'implementazione futura di interfacce con indirizzi di rete simili a 40.8.9.1 e 192.168.18.2.

Verifichiamo la corretta configurazione sul router con il comando *display interface Tunnel* 0/0/1 [figura 33].

```
(R8>dis interface Tunnel 0/0/1
Funnel0/0/1 current state : UP
Line protocol current state : UP
Last line protocol up time : 2021-05-11 23:44:04 UTC-08:00
Description:HUAWEI, AR Series, Tunnel0/0/1 Interface
Route Port, The Maximum Transmit Unit is 1500
Internet Address is 40.8.9.1/24
Encapsulation is TUNNEL, loopback not set
Tunnel source 192.168.83.1 (GigabitEthernet0/0/0), destination 192.168.59.2
Tunnel protocol/transport GRE/IP, key disabled
keepalive disabled
Checksumming of packets disabled
Current system time: 2021-05-11 23:44:56-08:00
    300 seconds input rate 0 bits/sec, 0 packets/sec
    300 seconds output rate 0 bits/sec, 0 packets/sec
    0 seconds input rate 0 bits/sec, 0 packets/sec
    0 seconds output rate 0 bits/sec, 0 packets/sec
    0 packets input,
                      0 bytes
    0 input error
    13 packets output,
                        1216 bytes
    0 output error
    Input bandwidth utilization
   Output bandwidth utilization : -
```

Figura 33. Visualizzazione dello stato e dei parametri configurati dell'interfaccia tunnel virtuale su R8.

Se nella prima e seconda riga dell'output lo stato è UP, significa che funziona correttamente. Inoltre, è possibile controllare anche altri parametri che sono stati configurati come l'IP address dell'interfaccia tunnel, l'indirizzo di sorgente e destinazione del tunnel, il tipo di protocollo che usa e altro che ora non andiamo ad elencare.

Possiamo anche vedere la tabella di routing. Infatti, nella figura 34 sarà evidenziata la rotta verso 192.168.92.0/24 che avrà come NextHop l'altro estremo del tunnel presente su R9 e che il traffico verrà instradato sull'interfaccia Tunnel 0/0/1 di R8.

Destination/Mask	Proto	Pre	Cost	Flags	NextHop	Interface
40.8.9.0/24	Direct	0	0	D	40.8.9.1	Tunne10/0/1
40.8.9.1/32	Direct	0	0	D	127.0.0.1	Tunne10/0/1
40.8.9.255/32	Direct	0	0	D	127.0.0.1	Tunne10/0/1
127.0.0.0/8	Direct	0	0	D	127.0.0.1	InLoopBack0
127.0.0.1/32	Direct	0	0	D	127.0.0.1	InLoopBack0
127.255.255.255/32	Direct	0	0	D	127.0.0.1	InLoopBack0
192.168.12.0/30	OSPF	10	3	D	192.168.83.2	GigabitEthernet
0/0/0						
192.168.14.0/30	OSPF	10	3	D	192.168.83.2	GigabitEthernet
0/0/0						
192.168.18.0/24	Direct	0	0	D	192.168.18.2	GigabitEthernet
0/0/1						
192.168.18.2/32	Direct	0	0	D	127.0.0.1	GigabitEthernet
0/0/1						
192.168.18.255/32	Direct	0	0	D	127.0.0.1	GigabitEthernet
0/0/1						
192.168.25.0/30	OSPF	10	4	D	192.168.83.2	GigabitEthernet
0/0/0						
192.168.31.0/30	OSPF	10	2	D	192.168.83.2	GigabitEthernet
0/0/0						
192.168.34.0/30	OSPF	10	2	D	192.168.83.2	GigabitEthernet
0/0/0						
192.168.42.0/30	OSPF	10	3	D	192.168.83.2	GigabitEthernet
0/0/0						
192.168.45.0/30	OSPF	10	3	D	192.168.83.2	GigabitEthernet
0/0/0						
192.168.59.0/30	OSPF	10	4	D	192.168.83.2	GigabitEthernet
0/0/0						
192.168.83.0/30	Direct	0	0	D	192.168.83.1	GigabitEthernet
0/0/0						-
192.168.83.1/32	Direct	0	0	D	127.0.0.1	GigabitEthernet
0/0/0						
192.168.83.3/32	Direct	0	0	D	127.0.0.1	GigabitEthernet
0/0/0						
192.168.92.0/24	OSPF	10	1563	D	40.8.9.2	Tunne10/0/1

Figura 34. Verifica della presenza nella tabella di routing della rotta verso la rete Customer 2 che sfrutta il tunnel GRE su r8.

Come ultima verifica, possiamo usare Wireshark per valutare il traffico presente sull'interfaccia di uscita di R8 (GigabitEthernet 0/0/0), generato da un'operazione di ping tra PC1 e PC2 [figura 35].

	icmp													
No		Time	Source	Destination	Protocol	Length	Info							
Г	29	64.218000	192.168.18.254	192.168.92.254	ICMP	98	Echo	(ping)	request	id=0x1ca9,	seq=1/256,	ttl=127	(no response '	found!)
	30	66.187000	192.168.18.254	192.168.92.254	ICMP	98	Echo	(ping)	request	id=0x1ea9,	seq=2/512,	ttl=127	(reply in 31)	
-	31	66.234000	192.168.92.254	192.168.18.254	ICMP	98	Echo	(ping)	reply	id=0x1ea9,	seq=2/512,	ttl=127	(request in 3	0)
	32	67.250000	192.168.18.254	192.168.92.254	ICMP	98	Echo	(ping)	request	id=0x1fa9,	seq=3/768,	ttl=127	(reply in 33)	
	33	67.296000	192.168.92.254	192.168.18.254	ICMP	98	Echo	(ping)	reply	id=0x1fa9,	seq=3/768,	ttl=127	(request in 3	2)
	34	68.312000	192.168.18.254	192.168.92.254	ICMP	98	Echo	(ping)	request	id=0x20a9,	seq=4/1024	ttl=127	(reply in 35)
	35	68.359000	192.168.92.254	192.168.18.254	ICMP	98	Echo	(ping)	reply	id=0x20a9,	seq=4/1024	, ttl=127	/ (request in :	34)
	36	69.375000	192.168.18.254	192.168.92.254	ICMP	98	Echo	(ping)	request	id=0x21a9,	seq=5/1280	ttl=127	(reply in 37)
	37	69.421000	192.168.92.254	192.168.18.254	ICMP	98	Echo	(ping)	reply	id=0x21a9,	seq=5/1280	, ttl=127	' (request in 3	36)
>	Fran	ne 30: 98 b	oytes on wire (784 bits),	98 bytes captured (78	4 bits)	on inter	rface	-, id	0					
>	Ethe	ernet II, S	Src: HuaweiTe_73:76:05 (00	:e0:fc:73:76:05), Dst	: Huawei	Te_86:0a	a:45 i	(00:e0:	fc:86:0a:	45)				
>	Inte	ernet Proto	ocol Version 4, Src: 192.1	68.83.1, Dst: 192.168	.59.2									
>	Generic Routing Encapsulation (IP)													
>	Internet Protocol Version 4, Src: 192.168.18.254, Dst: 192.168.92.254													
>	Inte	ernet Contr	rol Message Protocol											

Figura 35. Cattura del traffico generato dal ping tra PC1 e PC2 per verificare l'utilizzo del tunnel GRE tra le reti Customer.

Il frame contenente il pacchetto ICMP viene mandato da PC1 verso il router R8, poiché non conosce la rete 192.168.92.0 di cui fa parte PC2. Poi, R8 seguirà le indicazioni presenti sulla sua tabella di routing e veicolerà il pacchetto verso l'interfaccia Tunnel configurata. Il router sa che deve usare GRE, perciò quello che fa è mettere al pacchetto originario contenente ICMP un header GRE e un altro header IP che avrà come source e destination address del tunnel. Quanto appena detto, lo si può notare nella metà inferiore della figura: vediamo un header IP con source 192.168.83.1 (R8) e destination 192.168.59.2 (R9) corrispondenti agli indirizzi degli estremi del tunnel, un header GRE, l'header IP del pacchetto originale che ha come source PC1 e come destination PC2 ed infine abbiamo ICMP.

Infine, è interessante notare anche il processo OSPF 2 nell'interfaccia di R8 che funge da estremo del tunnel [figura 36].

No.	Time	Source	Destination	Protocol	Length	Info
3	6 79.953000	40.8.9.2	224.0.0.5	OSPF	106	Hello Packet
3	7 83.796000	192.168.83.2	224.0.0.5	OSPF	82	Hello Packet
3	8 84.640000	192.168.83.1	224.0.0.5	OSPF	82	Hello Packet
3	9 85.859000	40.8.9.1	224.0.0.5	OSPF	106	Hello Packet
- 4	0 89.906000	40.8.9.2	224.0.0.5	OSPF	106	Hello Packet
4	1 92.906000	192.168.83.2	224.0.0.5	OSPF	82	Hello Packet
4	2 93.812000	192.168.83.1	224.0.0.5	OSPF	82	Hello Packet
4	3 95.296000	40.8.9.1	224.0.0.5	OSPF	106	Hello Packet
4	4 99.843000	40.8.9.2	224.0.0.5	OSPF	106	Hello Packet
4	5 102.000…	192.168.83.2	224.0.0.5	OSPF	82	Hello Packet
4	6 102.953	192.168.83.1	224.0.0.5	OSPF	82	Hello Packet
4	7 104.734…	40.8.9.1	224.0.0.5	OSPF	106	Hello Packet
4	8 109.796…	40.8.9.2	224.0.0.5	OSPF	106	Hello Packet
4	9 111.109	192.168.83.2	224.0.0.5	OSPF	82	Hello Packet
5	0 112.140	192.168.83.1	224.0.0.5	OSPF	82	Hello Packet
5	1 114.171	40.8.9.1	224.0.0.5	OSPF	106	Hello Packet
5	2 119.765	40.8.9.2	224.0.0.5	OSPF	106	Hello Packet
5	3 120.218	192.168.83.2	224.0.0.5	OSPF	82	Hello Packet
54	4 121.296	192.168.83.1	224.0.0.5	OSPF	82	Hello Packet
5	5 123.625	40.8.9.1	224.0.0.5	OSPF	106	Hello Packet

ightarrow Frame 36: 106 bytes on wire (848 bits), 106 bytes captured (848 bits) on interface -, id 0

Ethernet II, Src: HuaweiTe_86:0a:45 (00:e0:fc:86:0a:45), Dst: HuaweiTe_73:76:05 (00:e0:fc:73:76:05)
 Internet Protocol Version 4, Src: 192.168.59.2, Dst: 192.168.83.1

> Generic Routing Encapsulation (IP)

> Internet Protocol Version 4, Src: 40.8.9.2, Dst: 224.0.0.5

> Open Shortest Path First

Figura 36. Cattura del traffico OSPF su R8 evidenziando la struttura dell'Hello Packet di OSPF 2.

Guardando la prima riga di cattura, possiamo vedere che R8 riceve un Hello Packet dall'altra interfaccia Tunnel configurata su R9 (40.8.9.2). L'Hello Packet è indirizzato a tutti i router che fanno parte del processo OSPF 2, poiché abbiamo come indirizzo di destinazione 224.0.0.5 (*AllSPFRouters*). Nella parte bassa della figura, possiamo vedere che al pacchetto originale è stato aggiunto l'header GRE e l'header IP del tunnel.

7.6. Configurazione GRE over IPsec

Per garantire maggiore sicurezza al traffico delle due reti Customer, si è deciso usare le funzioni di autenticazione e crittografia. Per ottenere queste proprietà, useremo GRE over IPsec, che consiste nell'usare il Tunnel GRE all'interno di un Tunnel IPsec, permettendo di avere riservatezza della comunicazione e di trasportare qualsiasi tipo di protocollo (usare solo IPsec permette di trasportare soltanto IP).

Prima di procedere alla configurazione di IPsec, bisogna selezionare il traffico interessante su cui implementare autenticazione e riservatezza e per farlo useremo ACL (*Access Control List*), che permette di filtrare e gestire il flusso di traffico [figura 37].



Figura 37. Creazione su R8 di ACL per filtrare il traffico interessante configurato.

Abbiamo creato un ACL con ID 3000 (da 3000 a 3999 sarà di tipo Advanced, poiché IPsec supporta solo questo tipo), dove è stata dichiarata una regola che impone di permettere il flusso di traffico GRE che ha come sorgente e destinazione gli estremi del Tunnel GRE impostato nel capitolo precedente. La stessa procedura fatta su R8 è fatta anche sul router R9, scambiando gli indirizzi di source e destination. Con il comando *display acl all* possiamo visualizzare tutte le regole configurate (in questo caso solo una) [figura 38].

```
<R8>dis acl all
Total quantity of nonempty ACL number is 1
Advanced ACL 3000, 1 rule
Acl's step is 5
rule 5 permit gre source 192.168.83.0 0.0.0.3 destination 192.168.59.0 0.0.0.3
```

Figura 38. Visualizzazione delle ACL e le sue relative informazioni su R8.

Una volta selezionato il traffico a cui applicare IPsec, configuriamo l'*IPsec Proposal*, che definisce i parametri di sicurezza per la negoziazione *IPsec SA (Security Association)*, che comprendono il protocollo di sicurezza, l'algoritmo di autenticazione, l'algoritmo di crittografia e la modalità di incapsulamento del flusso dati da proteggere. I peer su entrambe le estremità del tunnel (R8 e R9) devono avere la stessa IPsec Proposal per creare una connessione cifrata. Vediamo la configurazione in R8 nella figura 39.



Figura 39. Configurazione su R8 dell'IPsec Proposal scegliendo gli algoritmi di autenticazione e cifratura.

Abbiamo creato la Proposal con il nome *tran1*, che ci permette anche nella view di interfaccia dell'IPsec proposal in questione, dove abbiamo definito l'algoritmo di autenticazione e di cifratura. Per l'autenticazione è stato scelto SHA2 (ha un digest di 256 bit), poiché è la versione più comune di SHA. Infatti, il NIST (National Institute of Standards Technology) lo raccomanda al posto di MD5 o SHA1, poiché più sicuro in termini di resistenza alle collisioni. Però, il vantaggio di avere maggiore sicurezza lo paghiamo con la velocità di calcolo, poiché SHA2 è circa il 20-30% più lento rispetto al suo predecessore. Per la cifratura, invece, si è scelto AES-128 (chiave di 128 bit), scartando DES o 3DES poiché sono ormai troppo vulnerabili. Riteniamo soddisfacente la sicurezza offerta da AES-128, che ci dà il vantaggio di usare meno risorse computazionali rispetto alle AES con chiavi più lunghe. Se non specificato durante la configurazione, l'IPsec Proposal adotterà di default la modalità di incapsulamento di tipo tunnel, che è quella che vogliamo. Con il comando *display ipsec proposal* possiamo subito visualizzare tutte le sue caratteristiche [figura 40].

<r8>dis ipsec p</r8>	proposa	al	
Number of propo	sals:	1	
IPSec proposal	name:	tranl	
Transform	:	esp-new	
ESP protocol	:	Authentication	SHA2-HMAC-256
		Encryption	AES-128

Figura 40. Visualizzazione delle informazioni delle IPsec Proposal presenti su R8.

Passiamo ora alla configurazione dell'*IPsec Policy* che definisce il protocollo di sicurezza, l'algoritmo di autenticazione, l'algoritmo di crittografia e la modalità di incapsulamento facendo riferimento all'IPsec Proposal concordato tra i due peer. Prima di mostrare la configurazione, è importante far presente che i parametri *tunnel local, tunnel remote, inbound* e *outbound* presenti nella policy devono essere speculari sui router di interesse R8 e R9. Tenendo bene a mente questo, mostriamo la procedura di configurazione prendendo come esempio R8 [figura 41].



Figura 41. Creazione e configurazione manuale della IPsec Policy su R8.

Abbiamo creato la policy P1 (10 è il *sequence number*) e abbiamo optato per la configurazione manuale. Per questo motivo, oltre allo a selezionare il traffico interessante specificando l'ACL e dichiarare la proposta IPsec, dovremo configurare manualmente altri parametri relativi alla policy. Partiamo con il definire gli endpoint del tunnel, poi gli SPI (tag identificativo che consente di identificare un data flow) ed infine le chiavi di autenticazione. Ricordiamo che questi ultimi parametri dovranno essere specchiati sull'altro router CE R9.

Una volta creata la policy, bisogna associarla all'interfaccia che rappresenta l'estremo del tunnel crittografico come mostrato in figura 42.



Figura 42. Applicazione della policy P1 sull'interfaccia di R8 in cui vogliamo implementare il tunnel crittografico.

Con questo abbiamo concluso l'implementazione di IPsec. Ora, vediamo le differenze di configurazione tra R8 e R9 utilizzando il comando *display ipsec policy* su entrambi i router [figura 43].

<r8>dis ipsec policy</r8>	<r9>dis ipsec policy</r9>					
IPSec policy group: "Pl" Using interface: GigabitEthernet0/0/0	IPSec policy group: "P1" Using interface: GigabitEthernet0/0/1					
Sequence number: 10 Security data flow: 3000 Tunnel local address: 192.168.83.1 Tunnel remote address: 192.168.59.2 Qos pre-classify: Disable Proposal name:tranl Inbound AH setting: AH SPI: AH string-key: AH authentication hex key: Inbound ESP setting: ESP SPI: 12345 (0x3039) ESP string-key: huawei ESP encryption hex key: ESP authentication hex key: Outbound AH setting: AH SPI: AH string-key: AH string-key: AH authentication hex key: Outbound ESP setting: ESP SPI: 54321 (0xd431) ESP string-key: huawei ESP encryption hex key: ESP authentication hex key: ESP authentication hex key: ESP string-key: huawei ESP encryption hex key: ESP authentication hex key:	Sequence number: 10 Security data flow: 3000 Tunnel local address: 192.168.59.2 Tunnel remote address: 192.168.83.1 Qos pre-classify: Disable Proposal name:tranl Inbound AH setting: AH SFI: AH string-key: AH authentication hex key: Inbound ESF setting: ESF SFI: 54321 (0xd431) ESF string-key: huawei ESF authentication hex key: Outbound AH setting: AH SFI: AH string-key: AH authentication hex key: Outbound ESF setting: ESF SFI: 12345 (0x3039) ESF string-key: huawei ESF encryption hex key: ESF authentication hex key: ESF string-key: huawei ESF string-key: huawei ESF encryption hex key: ESF authentication hex key:					

Figura 43. Visualizzazione di tutti parametri della IPsec Policy su R8 e R9 e dell'interfacce su cui sono state implementate.

Infine, usiamo ancora una volta Wireshark per verificare che il traffico tra le due reti Customer venga protetto da IPsec. Nella figura 44, vedremo una cattura del traffico sull'interfaccia GigabitEthernet 0/0/0 di R8 durante l'esecuzione di un ping (avvenuto con successo) tra PC1 e PC2.

No.		Time	Source	Destination	Protocol Le	ength	Info
	31	71.078000	192.168.83.2	224.0.0.5	OSPF	82	Hello Packet
	32	73.265000	192.168.83.1	224.0.0.5	OSPF	82	Hello Packet
٠	33	78.047000	192.168.83.1	192.168.59.2	ESP	166	ESP (SPI=0x0000d431)
	34	79.859000	192.168.59.2	192.168.83.1	ESP	166	ESP (SPI=0x00003039)
	35	81.828000	192.168.83.2	224.0.0.5	OSPF	82	Hello Packet
	36	82.437000	192.168.83.1	224.0.0.5	OSPF	82	Hello Packet
	37	87.781000	192.168.83.1	192.168.59.2	ESP	166	ESP (SPI=0x0000d431)
	38	87.984000	192.168.83.1	192.168.59.2	ESP	166	ESP (SPI=0x0000d431)
	39	89.297000	192.168.59.2	192.168.83.1	ESP	166	ESP (SPI=0x00003039)
	40	89.969000	192.168.83.1	192.168.59.2	ESP	166	ESP (SPI=0x0000d431)
	41	90.015000	192.168.59.2	192.168.83.1	ESP	166	ESP (SPI=0x00003039)
	42	91.047000	192.168.83.1	192.168.59.2	ESP	166	ESP (SPI=0x0000d431)
	43	91.109000	192.168.59.2	192.168.83.1	ESP	166	ESP (SPI=0x00003039)
	44	91.594000	192.168.83.1	224.0.0.5	OSPF	82	Hello Packet
	45	92.140000	192.168.83.1	192.168.59.2	ESP	166	ESP (SPI=0x0000d431)
	46	92.219000	192.168.59.2	192.168.83.1	ESP	166	ESP (SPI=0x00003039)
	47	92.578000	192.168.83.2	224.0.0.5	OSPF	82	Hello Packet
	48	93.234000	192.168.83.1	192.168.59.2	ESP	166	ESP (SPI=0x0000d431)
>	Fram	e 37: 166	bytes on wire (1328 bits)	, 166 bytes captured	(1328 bits) on	interface -, id 0
>	Ethe	rnet II, S	Src: HuaweiTe_73:76:05 (00	:e0:fc:73:76:05), Dst	: HuaweiTe	86:0	a:45 (00:e0:fc:86:0a:45)
>	Inte	rnet Proto	ocol Version 4, Src: 192.1	68.83.1, Dst: 192.168	.59.2	-	
\sim	Enca	psulating	Security Payload				
	E	SP SPI: 0>	<0000d431 (54321)				
	E	SP Sequence	e: 520093696				
	> [Expected S	SN: 503316481 (16777215 SN:	s missing)]			
	1	Previous F	rame: 331				

Figura 44. Cattura del traffico su GE 0/0/0 di R8 dopo un ping tra PC1 e PC2 per verificare l'uso di IPsec.

Vediamo che l'unico traffico non protetto è quello del processo OSPF 1, cioè quello della rete ISP pubblica. Probabilmente, la serie di pacchetti protetti attraverso ESP al centro della figura 44 è relativa all'operazione di ping, cioè allo scambio di vari Echo Request e Echo Reply. Dagli indirizzi di sorgente e destinazione notiamo che il traffico viaggia trasportato dal tunnel crittografico (192.168.83.1 per R8 e 192.168.92.2 per R9), non rivelando nulla sui dati incapsulati e sulla loro origine. Quindi, possiamo concludere che il traffico aziendale tra le due reti dislocate viaggia in modo sicuro sulla rete pubblica.

8. Conclusione

Attraverso i capitoli precedenti abbiamo visto come eseguire la configurazione su un'intera topologia, pensata per avere una simulazione reale di una rete trattando le sue problematiche e le relative soluzioni da adottare. Inoltre, ha permesso di svolgere operazioni di riconfigurazione per adattarsi alle nuove situazioni ed esigenze della rete, che sono molto comuni ed essenziali oggigiorno. Un'altra opportunità fornita da questo progetto è quella di eseguire operazioni di troubleshooting, cioè di rivelazione di errori. Infatti, grazie ai comandi di display sui router e al software di cattura del traffico Wireshark. è stato possibile rivelare errori sulla configurazione e procedere in modo mirato alla loro risoluzione. Questi strumenti sono stati utili anche per la comprensione totale del funzionamento della rete IP, di come il traffico viaggia su di essa e delle funzioni dei primi layer dello modello ISO/OSI. Attraverso queste operazioni di diagnostica, è stato possibile garantire l'operatività della rete. Oltre al soddisfacimento delle specifiche, si è cercato di ottimizzare le prestazioni e di facilitare implementazioni future date dall'esigenza di modifiche al comportamento della rete o dalla sua espansione. Per concludere, dal lato della sicurezza si è scelto di trovare un compromesso tra una comunicazione più sicura e una minore richiesta di risorse computazionali per implementarla. Per questo motivo, si è optato per l'uso di algoritmi che garantiscono una buona resilienza agli attacchi, richiedendo una potenza di calcolo non eccessiva.

Bibliografia

[1] Materiale del corso HCIA Routing & Switching (Huawei Certified ICT Associate)