RETI II

Data Center: Multipath

Nei data center viene supportato il multipath: un meccanismo di routing che prevede che ci siano più righe in tabella di instradamento per un'unica destinazione. Per decidere quale percorso scegliere (quale riga della tabella di instradamento fa matching) si sfrutta il protocollo ECMP.

Il traffico multipath viene sfruttato per:

(1) distribuire e bilanciare il traffico
(2) aumentare la fault tolerance

Il multipath consente di avere più next-hop per lo stesso prefisso nella tabella di instradamento e ha bisogno di un supporto da parte del kernel. Per ogni pacchetto il kernel decide l'entry che va usata, e il processo decisionale si può basare su diverse policy. Una tecnica ricorrente è quella di sfruttare hash-based policy al livello 3 e 4. Un esempio potrebbe esere: i pacchetti appartenenti alla stessa connessione fanno la stessa strada. Per i pacchetti appartenenti alla stessa connessione viene generato un hash, i pacchetti che corrispondono a quell'hash vengono inviati sulla stessa linea. Per più connessioni TCP distinte si usano più cammini.

Architettura di un data-center ad alto livello

Un data center ha questa struttura:

- Collegamenti multipli in fibra ottica verso internet
- **Fabric**: infrastruttura progettata per trasportare i pacchetti tra i server e tra i server e Internet.
- **Server Farm**: infrastruttura che ospita applicazioni e servizi

Nei data center esiste un traffico dei due tipi:

- tra server (est-ovest). I server devono essere interconnessi. Per sapere come interconnettere bisogna conoscere i flussi di traffico (vengono supportate in questo modo le architetture a microservizi, se si prevedono grandi flussi di dati verrà dedicata più banda).
- server-internet (nord-sud)

Che servizi offrono i data center?

Esistono tre diversi modelli di servizi:

- **Data center in sede**: costruiti e gestiti da un'azienda.
- **Data center co-locati**: costruiti da un'azienda, che come servizio offre dello spazio all'interno del suo data center. L'azienda che ospita gestisce l'infrastruttura (edificio, sicurezza, la larghezza di banda), mentre l'azienda che è ospitata porta i propri componenti (server, firewall ...)
- **Data center cloud:** costruiti da un'azienda, che come servizio offre dello spazio nel cloud.

Requisiti nella realizzazione di un data center

I requisiti di rete di un data center sono:

- **Supportare alta banda nelle comunicazioni server to server**. Applicazioni che si basano su calcoli in cluster, possono coinvolgere centinaia di server per raggiungere un risultato. Tutte le architettetture a microservizi si basano fortemente sulla comunicazione tra server. Si immagini uno scenario con più VM/containers distribuiti su più server devono scambiarsi una grande quantità di dati tra loro nel modo più efficiente possibile. Ogni VM vuole comunicare con gli altri host usando la massima banda possibile della propria scheda di rete (NICs). Se si disponde di una scheda a 100 gbps si vuole usare la banda massima a prescindere dalla situazione del traffico in quel momento nel data center (ossia senza preoccuparsi della congestione).
- **Supportare la scalabilità.** I data center variano da poche centinaia a centinaia di migliaia di server in un'unica sede fisica.

- **Supportare la resilienza.** Le applicazioni dei data center sono progettate per funzionare anche in presenza di guasti. Se ce n'è la necessità, dovrebbe essere possibile acquistare e distribuire hardware simile a quello già in uso.

I nodi di un data center possono essere connessi usando due tipi di compontenti:

- **Hardware specializzato.** Opzione costosa: gli over the top (es. Google) progettano hardware e lo fanno realizzare
- **Switch e router di largo consumo.** Opzione economica: vengono usati switch e router per scalare.

Topologia di un data center

Bisogna stabilire una tipologia che supporti le qualità richieste nei requisiti. Si fa notare che, mentre la rete internet cambia topologia e dimensioni in modo del tutto spontaneo, senza l'intervento di un ente centralizzato, in un data center è possibile scegliere la tipologia della rete.

È molto popolare la **tipologia Fat-Tree** un'architettura a tre livelli di nodi (tutti i nodi hanno le stesse caratteristiche), che consente di supportare ridondanza, scalabilità, la suddivisione in 2 costante della larghezza di banda disponibile.

Perché proprio un Fat-tree?

- Tree: In una tipologia ad albero (non Fat) le foglie hanno un solo cammino verso la radice: non va bene.
- Fat-Tree: preso un albero, ogni nodo viene espanso. In questo modo ci sono più cammini dalla foglia verso i tof.

Due parametri definiscono un Fat-Tree:

- **2K** = La radice di un nodo. Se la radice di un nodo è 2K, esso ha K connessioni verso il nord e K connessioni verso il sud. La radice indica in sostanza il numero di porte disponibili.
- **R** = fattore di ridondanza
- Quando K≠R il Fat-Tree è chiamato multiplane.

I nodi/ gruppi di nodi di un Fat-Tree vengono chiamati in questo modo:

- **Foglia:** nodo connesso con la server farm
- **Spine:** nodo che si trova in mezzo tra le foglie e i tof
- **Point of delivery (pod):** serie di foglie e spines completamente interconnesse
- **Top of fabric (tof):** nodi superiori che forniscono comunicazione tra i pod
- **Planes:** serie di tof

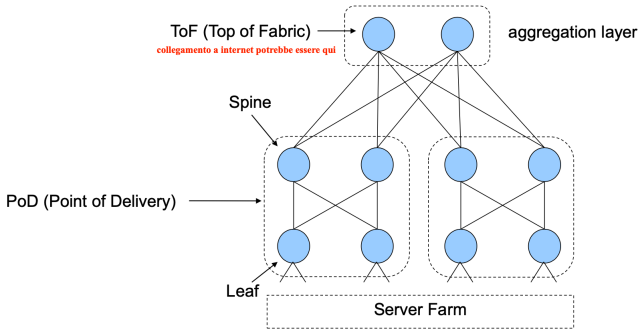

Figure 1. nodi/ gruppi di nodi di un Fat-Tree

I nodi sono router o switch? Dato che lo spanning tree consente un solo cammino per VLAN e invece si vuole usare multi-path si decide di usare i router.

I nodi che sono collegati a internet sono solitamente i tof oppure le foglie.

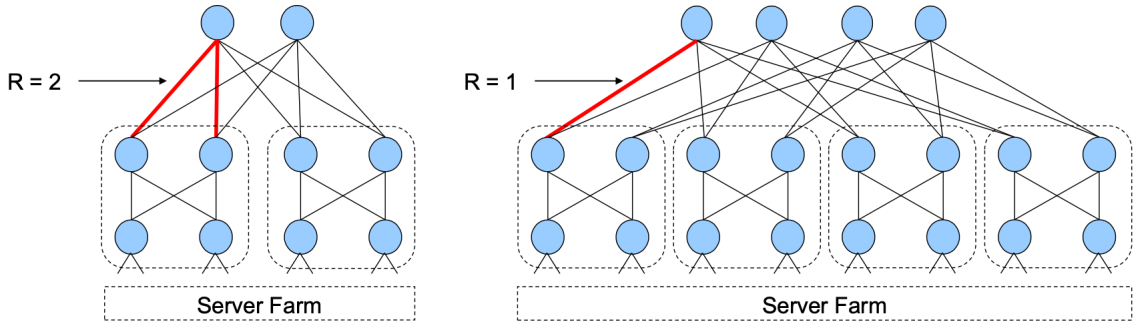

FIGURE 2. con R=1 sarebbe possibile aggiungere altri due TOF (per non perdere requisito di banda) ma in questo modo si ha solo un link che collega TOF a un POD (viene persa ridondanza). Per questo motivo viene preferita la configurazione di sinistra.

SCELTE DI ROUTING IN UN FAT-TREE

È stato già accennato il motivo per cui potrebbe essere più utile scegliere una tecnologia che lavora al livello 3 rispetto ad una che lavora al livello 2. Tuttavia è possibile approfondire più nel dettaglio quali sono le valutazioni che è utile formulare prima di prendere una decisione.

- Il vantaggio principale di un router è il supporto al multipath. Purtroppo però la configurazione di un router è più complessa (va automatizzata) e va scelto idoneo protocollo di routing (**viene scelto BGP**).
- Il vantaggio principale di uno switch risiederebbe nella facilità di configurazione di questi apparati. Purtroppo però gli switch:
  (1) non supportano il multipath (invece **BGP supporta nativamente il protocollo ECMP**).
  (2) non è facile fare bilanciamento di carico (**è estremamente semplice fare bilanciamento di carico con BGP**).
  (3) il traffico broadcast potrebbe inondare la rete.

BGP viene scelto nella sua forma eBGP. Il motivo è intuitivo: non si vuole introdurre un IGP che generi molto traffico broadcast. OSPF è necessario per il funzionamento di iBGP e fa selctive flooding per aggiornare il database di tutti i nodi. **Con BGP se la best rimane la stessa non viene propagata variazione della tipologia della rete**. Per evitare l'inondazione di traffico broadcast si rinuncia ad usare un IGP (e di conseguenza un iBGP) a favore di eBGP.

BGP usato in internet ha gli stessi obiettivi di BGP utilizzato in un data center? La risposta è no. In internet si desidera un BGP stabile, la perdita di pacchetti è consentita per un piccolo intervallo di tempo (non è una priorità che BGP converga nel modo più veloce possibile), e per ogni destinazione esiste un singolo percorso.

Nei data-center perdere pacchetti è malvisto, perché non si vuole che la lentezza della rete gravi in modo significativo sulle applicazioni e sui servizi. Per questo BGP deve convergere nel modo più rapido possibile.

Automazione della configurazione. Quando viene tirato su un peering viene specificato il nome dell'interfaccia anziché l'indirizzo IP e il numero di AS remoto.

Vengono creati gruppi di peer per specificare le politiche per gruppi.

BGP IN UN DATA CENTER

Gli AS number globali non vengono usati nei dati center, quindi si introducono gli AS number privati.

Idea: Ogni nodo (router) viene visto come un AS e ha un AS number privato che lo identifica.

Problema: Questo approccio porta a path exploration, fenomeno di instabilità nel routing.

Si immagini che avvenga un cambiamento della rete.

Un nodo sta ancora aspettando che arrivino tutti gli annunci aggiornati, nell'attesa il routing diventa inconsistente.Il router sta esplorando i vari percorsi possibili, ha molte alternative e passa da una all'altra prima che vengano ritirate tutte. In questo periodo transitorio si propagano nella rete percorsi incoerenti, e vengono trasmessi molti aggiornamenti BGP inutili. È un fenomeno normalissimo in BGP, e per risolverlo ci vogliono 4-5 minuti, tempo non accettabile in un data center. Inoltre le topologie dei data center sono molto dense e hanno molti cicli, proprio in tipologie di questo tipo il problema è più consistente e deve essere affrontato.

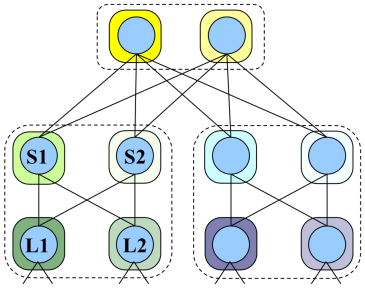

FIGURE 3. Path exploration issue

Path exploration issue:

(1) L2 va annunciare ai suoi vicini (S1, S2) il suo prefisso
(2) S1, S2 propagano l'annuncio all'interno del POD (entrambi mandano annuncio ad L1)
(3) L1 tiene entrambi gli annunci, uno sarà la best $>$ (esiste sempre e solo al massimo una best) esistono più rotte con $=$ (alternative alla best)
(4) L1 sceglie come la best S1 e la manda a S2
(5) S2 riceve annuncio per raggiungere L2 tramite L1 (ha AS-PATH S2 - L1 - S1 - L2). Non sarà la sua best perché ha AS PATH $>>$ rispetto all'annuncio mandato direttamente da L2
(6) **problema**: L2 si rompe
(7) S1-S2 si accorgono che l'interfaccia è down allora ritirano l'annuncio.
(8) S2 ha ancora una rotta (glielo ha mandato L1, che non ha ritirato l'annuncio), che diventa la best.
(9) S2 manda la best (withdrawal) a L1. Che succede quindi se arriva traffico da nord (TOF) per L2? Fa questo percorso: S2 - L1 - S1 (qui muore) però potrebbe finire in un loop. In entrambi i casi è traffico inutile e nocivo per il data-center.

Soluzione:

- le TOF nello stesso plane appartengono allo stesso AS.
- le spine nello stesso POD appartengono allo stesso AS.
- ogni foglia è un AS.

Non ci sono ritorni indietro, traffico non passa due volte sullo stesso link (NON c'é iBGP, due spine in un POD non si scambiano annunci)

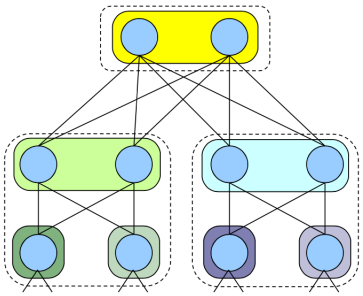

FIGURE 4. ASes scheme

Questo schema di AS permette 4 path per foglia, **ogni percorso ha uguale AS path** (ossia si rimane sempre all'interno dello stesso POD e dello stesso plane). BGP supporta nativamente ECMP. Per impostazione predefinita, BGP considera due annunci dello stesso prefisso "uguali" se sono uguali in ogni criterio di selezione del miglior percorso, tranne che nell'ultimo (router-id $<<$). Per abilitare il multipath, **BGP richiede che gli AS path selezionati per il multipath siano uguali.**

Questo sembra funzionare per lo schema in figura 4, ma potrebbe non è essere così in un Fat-Tree multiplane.

Allora viene introdotto **BGP multi-path relax**: basta che gli AS path abbiano la stessa lunghezza che il multipath può essere abilitato.

timers BGP

Esistono quattro timer principali che sono responsabili del comportamento di BGP

- advertisement timer
- timer di keepalive e timer di hold.
- timer di connessione.

Advertisement timer. Gli annunci che devono essere inviati a un vicino sono raggruppati e inviati insieme solo quando l'intervallo scade. Allora il timer viene azzerato per quel vicino.

Il valore predefinito per eBGP è di 30 secondi. Nel routing tra i diversi AS questo migliora la stabilità e riduce il numero di aggiornamenti multipli per lo stesso prefisso. Nei data center è impostato a 0s, è necessario per una rapida convergenza.

Timer di keepalive e di hold. Ogni peer BGP invia periodicamente messaggi keepalive ai propri vicini in base al keepalive timer.

Quando un peer non riceve un keepalive per un periodo superiore al timer di hold la connessione viene interrotta e tutti gli annunci ricevuti sono considerati non validi. Il peer tenta poi di ristabilire la connessione.

Per impostazione predefinita, il timer di keepalive è di 60s e il timer di hold è di 180s. Nei data center vengono utilizzati timer di 3s e 9s rispettivamente.

Timer di connessione. Quando la connessione con un peer fallisce, BGP attende la scadenza del timer di connessione prima di tentare una nuova connessione. Il timer di connessione per impostazione predefinita è 60s. Questo può ritardare il ristabilimento della sessione, nei data center è impostato a 10s.

Requisiti

Requisiti dei tenant. I requisiti dei tenant (customers, inquilini del data center) sono:

- Ogni tenant vuole gestire in modo indipendente il proprio spazio di indirizzi IP privati (potrebbero esistere piani di indirizzamento sovrapposti tra i vari customer).
- Il traffico dei container/VM deve essere segregato tra i tenant e tra il traffico del data center.

Requisiti di un'architettura server. I server devono supportare la virtualizzazione tramite container e VM, con l'obiettivo di quadagnare:

- alta disponibilità (garantita tramite **orchestrazione**).
- isolamento completo tra diversi container o VM sullo stesso server.
- flessibilità nell'assegnare container o VM a diversi tenant.

Requisiti di un orchestratore. Un orchestratore è un software che gestisce il ciclo di vita dei container e VM. Le responsabilità funzionali di un orchestratore sono:

- creare, distruggere i container e VM.
- spostare un container/VM da un server all'altro con la possibilità di mantenere le configurazioni di rete (MAC/IP) e garantendo tempi di inattività minimi.
- allocare in modo ottimale le risorse e la gestire i guasti.

I requisiti dei tenant, dei server, dell'orchestrazione fanno intuire come sia necessario l'utilizzo di tunnel.

Requisiti di un protocollo di tunneling.

- configurazione minima
- incapsulare il traffico di ciascun tenant (è necessario un identificatore del tenant)
- incapsulamento in Layer-4 per sfruttare il Multi-Path IP, per attraversare i router, per attraversare Internet. I diversi data center devono interconnettersi in modo trasparente via Internet (per creare le cosiddette regioni).

Le esigenze molto simili a quelle che avevano portato all'utilizzo delle VPN MPLS (IP in MPLS) ma non si vuole usare quel protocollo per non complicare la configurazione.

Inoltre non è possibile usare le VLAN, una tecnologia di livello 2

Allora si usano le VXLAN, create ad hoc per le esigenze di un data center. Le VXLAN prevedono un tunnel Layer 2 in Layer4.

Scenario senza le VXLAN

Situazione. I server collegati alle leaf dei vari PoD devono essere raggiungibili e visibili in rete rispettando i requisiti proposti.

Possibile idea.

- Ogni foglia annuncia il prefisso del server a cui è collegato.
- Ogni server ha un IP.

Svantaggi.

- Container e VM condividono lo stesso IP del server, non è possibile spostare un container da un server all'altro senza rimappare l'IP.
- I tenants devono seguire il piano IP del data center, senza avere la possibilità di esporre gli IP dei container.
- Non c'é isolamento del il traffico relativo a servizi di diversi tenants.

VXLAN, Virtual eXtensible Local Area Network

VXLAN è una tecnologia progettata per rispondere alla necessità delle reti all'interno di data center e prevede l'incapsulamento Ethernet in UDP. Un tenant, usando le VXLAN, ha a disposizione una rete ethernet distribuita in giro per il mondo

Ogni VXLAN è caratterizzata da:

- **Un VNI**: VXLAN Network Identifier.Un identificatore di uno specifico tunnel VXLAN, spazio di indirizzamento a 24 bit, più di 16 milioni di VNI possibili.
- **Più VTEP**: End Point del tunnel VXLAN (equivalente ai PE in MPLS). Dispositivo che incapsula e decapsula i pacchetti VXLAN.

La tabella MAC to VTEP. La tabella MAC to VTEP si trova in ogni VTEP, è simile alla tabella di inoltro degli switch. Per ogni VNI il VTEP mantiene una tabella di coppie <MAC,IP>, dove viene memorizzata la corrispondenza tra gli indirizzi MAC e gli IP del VTEP di destinazione. Ogni interfaccia fisica o VLAN di un VTEP è assegnata a un VNI. Gli indirizzi MAC appresi sulle *proprie* interfacce sono locali, e gli IP delle loro coppie sono sostituiti dalla parola *locale.*

Quando un VTEP riceve da un'interfaccia locale (assegnata a un VNI) un frame:

- Verifica l'esistenza, nella VNI, della coppia <indirizzo MAC destinazione del frame, IP della VTEP destinazione>.
- se la coppia esiste, il VTEP incapsula il frame e invia il pacchetto risultante all'IP VTEP di destinazione.
- se non esiste, il frame viene inviato a tutti gli altri VTEP (incapsulato) e a tutte le porte locali di quella VNI sperando che qualcuno risponda..

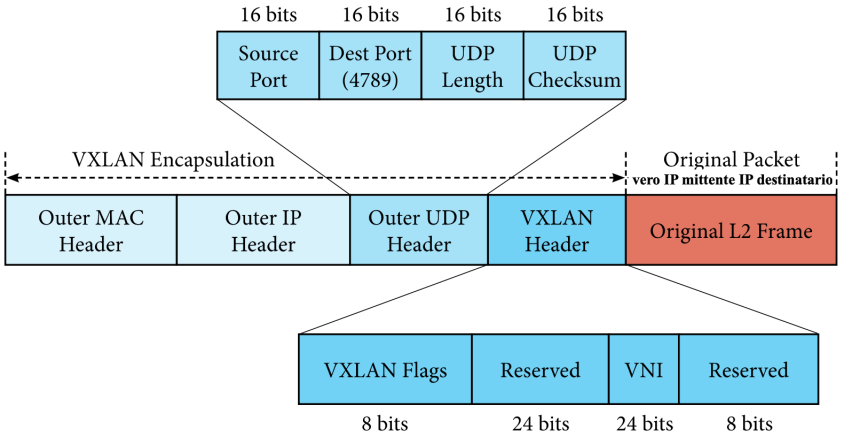

Figure 5. Struttura del pacchetto. Il pacchetto rosso viene imbustato in un pacchetto VXLAN, che viene poi imbustato in un pacchetto UDP, poi IP, poi Ethernet

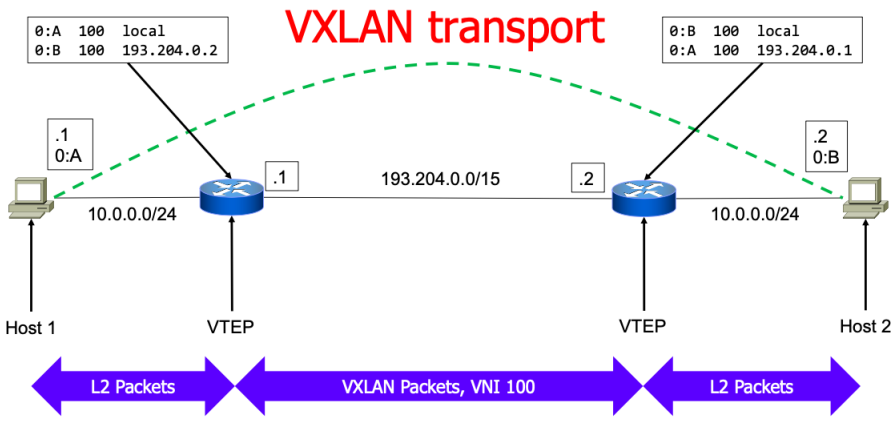

FIGURE 6. VXLAN transport. Un pacchetto parte dall'host 1 e vuole raggiungere l'host 2 che vede direttamente connesso. Nessuno dei due host sa che il pacchetto verrà incapsulato e potrebbe attraversare un numero indefinito di router sparsi in internet, prima di raggiungere la VTEP destinazione.

Percorso che fa il pacchetto VXLAN:.

(1) Da una VM a una VTEP**.** La VM invia il pacchetto rosso che contiene l'IP mittente-destinatario, MAC mttente-destinatario (anche se host non è d.c viene visto come tale, proprio perché si tratta di una VXLAN).
(2) Dalla VTEP a internet. La VTEP procede con l'incapsulamento. Vengono aggiunti VXLAN header, pacchetto liv 4, liv 3, liv 2.
   - **VXLAN header:** la VNI è nota grazie alla tabella MAC to VTEP che registra le corrispondenze indirizzo MAC con l'IP della VTEP destinazione. Il campo viene riempito sulla base del MAC destinazione del pacchetto rosso.
   - **UDP header:** quando una VTEP manda un pacchetto a un'altra VTEP lo fa sulla porta 4789. La source port viene cambiata randomicamente e sfruttata per calcolare l'hash usato per distinguire i diversi flussi nel multipath.
   - **IP header:** l'IP mittente è la VTEP mittente, l'IP destinazione è la VTEP destinazione.
   - **MAC header:** MAC mittente corrente e destinatario corrente.
(3) Da internet alla VTEP destinazione
(4) Dalla VTEP alla destinazione del pacchetto

Gestione del traffico broadcast. Traffico broadcast viene generato:

- da uno switch che ancora non ha fatto learning e quindi invia il traffico momentaneamente su tutte le sue porte.
- traffico ARP.

Nel caso delle VXLAN il traffico broadcast è legato al popolamento della tabella MAC to VTEP delle varie VTEP.

Come si popola la tabella MAC to VTEP se i MAC address sono remoti?

Idea. VXLAN utilizza IP multicast group, un router si può sottoscrivere a un gruppo IP multicast. Se passa un pacchetto con quell'IP, tutti i router subscribed prendono quel pacchetto. Il pacchetto ha un unico indirizzo destinazione ma arriva a n router. Si mappa per ogni VNI un gruppo multicast, e ogni VTEP si sottoscrive a quel gruppo multicast.

Svantaggi. Complessità nella configurazione, l'obbligo di fare il deploy di altri protocolli IGMP, IGMP Snooping...

Soluzione. La soluzione migliore è usare **EVPN-BGP**. EVPN BGP utilizza BGP MultiProtocol per trasportare gli indirizzi Ethernet come annunci. Con BGP MultiProtocol, BGP acquista la possibilità di gestire più address family (IPv4, IPv6, **L2VPN**). Quello che succede è che i MAC address dei VNI vengono annunciati dalle VTEP tramite BGP. Sul file di configurazione della VTEP è possibile scrivere "Dovrai mandare annunci L2VPN (annunci che contengono MAC address) al neighbor x"

In più le VTEP "proxano" (filtrano ?) traffico broadcast per evitare flooding.

Il peering tra le due VTEP è multihop, ossia le due VTEP possono essere lontane in modo arbitrario.

Osservazione: la VTEP si comporta da swtich su tutte le interfacce interne (quando dialoga con gli host fa learning di tutti i MAC address), si comporta da router sulle interfacce esterne (verso altre VTEP) che partecipano nel peering EBGP, fanno multipath, etc...

Osservazione tecnica: La necessità di impostare manualmente l'MTU (Maximum Transmission Unit) dei pacchetti che passano per le interfacce di dispositivo associato a una VNI. L'header VXLAN occupa 50 bytes, quindi l'mtu del pacchetto Ethernet diminuisce da 1500 a 1450 bytes.

Configurazione di una VTEP.

- A livello tecnico una VTEP è un driver che mette il pacchetto dentro l'envelope (e decapsula pacchetti).
- **Uso di un bridge virtuale.** È possibile utilizzare tanti bridge virtuali quante sono le VNI, oppure un solo bridge virtuale che associ a tutte le VTEP tutte le porte. La funzione del bridge è fare da ponte tra le **porte** e le relative **VTEP**. È utilizzato dal VTEP per fare learning in combinazione con EVPN-BGP (il bridge si occupa della interfacce locali, EVPN-BGP dei MAC address remoti). La tabella di inoltro del bridge viene popolata anche attraverso gli aggiornamenti ricevuti da BGP. Il control plane di FRR guarda agli aggiornamenti della tabella del bridge per inviare aggiornamenti via BGP.
- **Uso di VLAN.** Sulla VTEP ci sono tante VLAN, una VLAN serve a non mischiare i MAC address dei tenants. Si mappa per ogni VNI un ID VLAN. VNI permette di scalare, ma l'ID VLAN è limitato superiormente da 4096. Usare le VLAN riduce la possibilità di scalare? Risposta: No. Una VLAN identifica il tenant nel collegamento tra il server e la leaf, mentre la VXLAN è univoca nel data center. Si presume che un singolo server non ospiti mai più di 4096 tenants, quindi le VLAN non rappresentano un collo di bottiglia. Per uno stesso tenant in LAN diverse del data center possono corrispondere VLAN diverse, ma sempre la stess VXLAN.
- **Qual è il compito di una foglia in un data center?** Una foglia ospita una VTEP, toglie i pacchetti dall'incapsulamento VXLAN e li mette in incapsulamento VLAN, in modo da distinguere i diversi tenant all'interno del server. Il server utilizza gli ID VLAN per inoltrare i pacchetti ai contenitori/VM corretti. Il server leva il tag dai pacchetti in modo che i container/VM non siano a conoscenza delle VLAN.

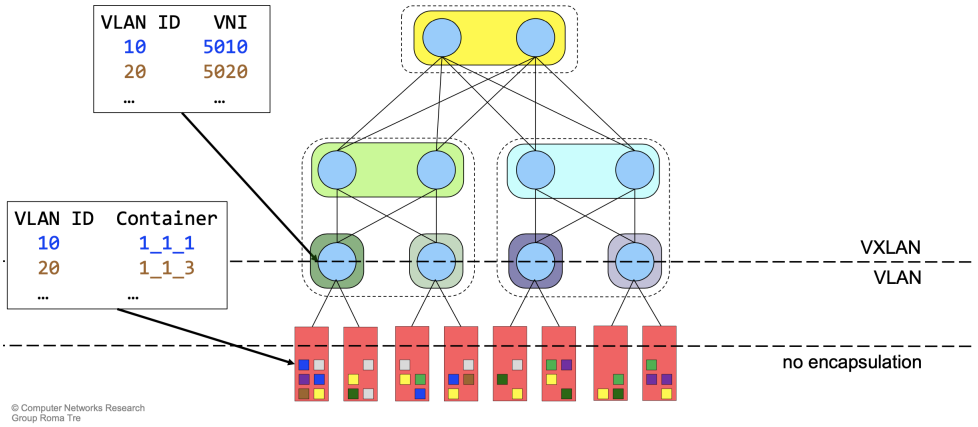

FIGURE 7. VXLAN and VLAN scheme

Come è possibile aggiungere ridondanza in modo che, se una foglia si rompe, il server ad essa collegata non vada giù? Aggrega più NICs (interfacce) in un'unica interfaccia logica. In questo modo se un link va giù c'è il secondo link attivo.

Le leaf dello stesso PoD hanno stesso IP sulle loopback (anycast). La tecnica anycast permette a più macchine sparse in tutto il mondo di annunciare lo stesso prefisso, gli host raggiungono la macchina che gli conviene. Le leaf annunciano la stessa interfaccia di loopback, i vari server decidono quale link attraversare in base alla convenienza, o in base alla policy in quel momento attiva.

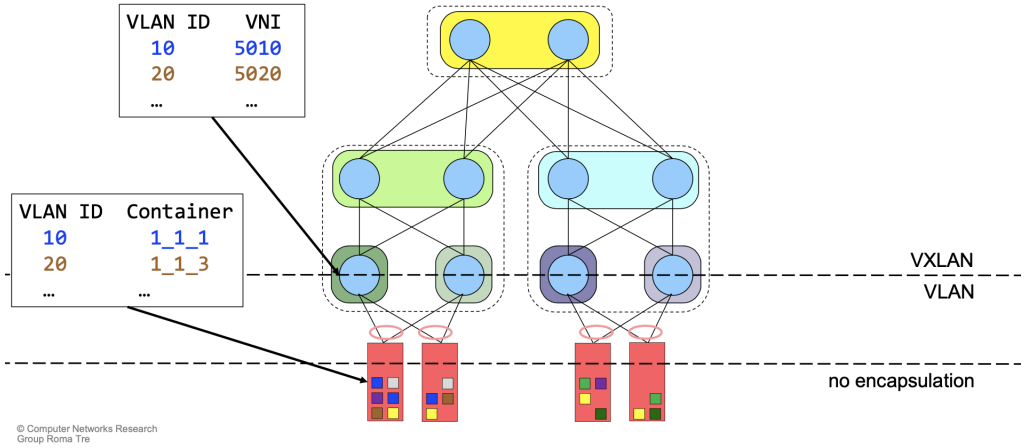

FIGURE 8. più NICs aggregate in un'unica interfaccia logica

Sui 2 link per la ridondanza vengono solitamente applicate politiche di load balancing. Un esempio è il protocollo 802.3ad. Il protocollo è di tipo active-active (entrambi i link attivi) e lavora con un hash. Per supportare il doppio collegamento in modalità active-active è necessaria la Multi-Chassis Link Aggregation (MLAG). MLAG consente a un server di collegare due porte a **diversi** switch e di operare come se fossero collegate ad un unico switch logico. In questo modo si ottiene una maggiore ridondanza e un maggiore throughput del sistema. I kernel Linux classici non supportano MLAG

alternative per progettare un servizio Web scalabile

Il contesto. Il servizio Web è replicato su più server distribuiti geograficamente, ad esempio in vari data center. A seconda del meccanismo di distribuzione i server potrebbero avere:

- diversi nomi, diversi indirizzi IP
- un solo nome, diversi indirizzi IP
- un solo nome, un solo indirizzo IP

I meccanismi di distribuzione:

- il meccanismo di name resolution è gestito dal DNS al livello di sessione. Cioé la name resolution restituisce un IP che rimarrà lo stesso per tutta la sessione.
- anycast BGP è gestito dal router al livello di sessione.
- HTTP redirection è gestito dal web server al livello di page request.
- La packet redirection è gestito dal load balancer al livello di granularità che più si preferisce.

meccanismo	entità	granularità
name resolution	DNS	sessione
anycast BGP	router / routing	sessione
HTTP redirection	Web server	page request
packet redirection	locale Load Balancer	sessione / page request / hit

Table 1. tassonomia dei meccanismi

I meccanismi di mirroring (che non compare in tabella), name resolution, anycast BGP e HTTP si chiamano **meccanismi di distribuzione globale,** perché l'intera architettura non è situata in una singola locazione fisica (in un unico Data Center) ma distribuita in tutto internet. Name resolution, HTTP redirection e packet redirection sono **meccanismi di distribuzione locale,** perché potrebbero svolgere le proprie funzioni anche solo limitatamente ad un data center

Mirroring

Cos'è. Le informazioni/risorse sono replicate su siti Web geograficamente distribuiti e vengono offerti diversi nomi, diversi indirizzi IP.Un utente che per accedere a un sito può scegliere tra le varie repliche:

- HTTP://cygwin.com
- HTTP://mirrors.163.com/cygwin/(China)
- HTTP://mirror.isoc.org.il/pub/cygwin/(Israel)

Pro e contro. Si tratta di un'architettura semplice, ma la replicazione è visibile per l'utente e non si ha distributzione di carico controllata.

HTTP redirection

Cos'è. Fa parte di HTTP standard, supportato dai browser.

Esempio:

- Un client ha fatto il three way handshake con un server, e procede con una GET.
- la richiesta viene riassegnata su un secondo server. Il client finisce la connessione con il primo (FIN, ACK, FIN,ACK) e per la seconda volta procede con il three way handshake, questa volta con il secondo server.
- Il client fa una GET. Questa volta il secondo server gli restituisce la risorsa.

A differenza del mirroring è trasparente all'utente. (non al client browser). La ridirezione può essere verso: un indirizzo IP ,un nome.

Il meccanismo può essere attivato in modo centralizzato (uno specifico server distribuisce il carico) oppure distribuito (qualunque server, magari quando è sovraccarico). Si può fare una selezione sulle pagine di cui si vuole effettuare il redirect.

Pro e contro. Questo meccanismo di scheduling è semplice ma ha degli aspetti negativi: consente di ridirigere solo le richieste HTTP, può aumentare traffico e tempo di risposta. Infatti il client instaura due connessioni HTTP per ogni richiesta.

Name resolution

Ripasso su come avvenga la risoluzione dei nomi. Il resolver di un host interroga il proprio name server di default su quale sia l'indizzo IP che corrisponde a sally. ee. berkeley. edu:

(1) il name server si rivolge al name server radice (autorità su tutti)
(2) che a sua volta rimanda al name server edu (si inizia a riscendere lungo l'albero)
(3) che a sua volta rimanda al name server berkeley.edu
(4) che a sua volta rimanda al name server nameserver.berkeley.edu, name server autorità per sally. ee. berkeley. edu
(5) finalmente viene risolto il nome e trovato l'IP

Le informazioni DNS sono memorizzate in record. Ogni dominio (anche un singolo host) ha associato un resource record.

Esempio:
Qui è stato trovato l'IP del nameserver autorità.

- berkeley.edu NS nameserver.berkeley.edu
- nameserver.berkeley.edu IN A 50.50.50.50

Qui è stato trovato l'IP di interesse.

- sally.berkeley.edu IN A 70.70.70.70

Scheduling nella name resolution, cos'è. Il meccanismo di scheduling viene effettuato nella fase di lookup della risorsa e permette di:

- assegnare un diverso IP a seconda della provenzienza della richiesta.Il bilanciamento viene fatto sulla base dell'area geografica. Si preferirebbe infatti assegnare ad un client un server "vicino".
- assegnare un diverso IP (anche se i client si trovano localmente nella stessa zona) in modo da bilanciare il carico sui diversi server.

Come è possibile fare bilanciamento? Per uno stesso nome, c'è un pool di indirizzi IP in posti diversi del mondo che permettono di accedere alle stesse risorse.

Ruolo del TTL nello scheduling. Il client chiede l'indirizzo IP corrispondente al nome del sito. Riceve uno specifico indirizzo IP e un opportuno TTL. Quando il TTL scade, il client deve chiedere di risolvere di nuovo il nome al DNS e potrebbe essergli assegnato un server diffrente (per motivi di load balancing).

TTL costante. Una possibilità è settare TTL con valori molto bassi, per avere pieno controllo dello scheduling da parte del DNS. Si rischia di sovraccaricare il name server autorità del sito e le cache dei browser spesso ignorano il TTL.

TTL adattativo. Viene modificato dinamicamente il TTL in base al carico sui server, alla frequenza di richieste da un certo dominio.

Difficoltà. A causa del caching, i DNS controllano solo una parte del traffico destinato ad un certo sito. Ridurre di molto il TTL pone vari problemi: il TTL spesso non ha effetto sulla cache dei browser e i DNS non cooperativi ignorano i TTL molto piccoli.

IP anycast

Cos'è. Un meccanismo per cui lo stesso prefisso viene annunciato da più AS geograficamente distribuiti in Internet. Ogni AS che annuncia il prefisso contiene una replica del servizio Web. Il BGP di ogni AS in Internet sceglie l'annuncio che corrisponde alla metrica migliore (ad es. all'AS-PATH più breve). Al client viene offerto un solo nome, un solo IP.

Algoritmi di scheduling

Come scegliere il server che risponderà alla richiesta del client? Mentre nel mirroring e con IP anycast non è possibile selezionare il server (in un caso ci pensa l'utente, nell'altro BGP), nei meccanismi HTTP redirect e DNS c'è più flessibilità. Una buona allocazione del client al server tiene conto del carico del server e della prossimità tra client e server.

È possibile utilizzare **algoritmi statici** come round robin o funzioni hash, oppure **algoritmi dinamici** sfruttando le informazioni sul client e lo stato dei server.

Una tecnica per stimare la prossimità di un client da un server è utilizzare l'indirizzo IP (in che AS si trova, informazioni sull'AS) o il numero di hop che li separano (traceroute). La difficoltà sta nel fatto che i link non hanno tutti la stessa banda e quindi più che la prossimità geografica sarebbe utile conoscere la latenza (il tempo impiegato dal client per inviare una richiesta e ricevere la risposta dal server). Per trovare la latenza è possibile utilizzare il comando PING o un emulatore di richieste HTTP.Ovviamente per fare questa stima si introduce un'overhead.

Scheduling gerarchico. Consiste nel far uso di più meccanismi di scheduling in successione. Si parla di scheduling a due o tre livelli... (scheduling basato su DNS, poi su HTTP redirect).

PACKET REDIRECTION

Se il meccanismo di scheduling è locale vuol dire che l'intera architettura (i server, macchine coinvolte nel meccanismo) è situata in una singola locazione fisica. I server possono essere macchine fisiche o virtuali a cui viene assegnato un VIP.

virtual IP. un IP che rappresenta diversi server e un URL www.dominio.com usata per risolvere i nomi . Il load balancer è un componente che smista le richieste (effettua il mapping tra VIP e indirizzo dei server). I load balancer possono eseguire un controllo fine sulle richieste a livello di hit, page request o sessione.

Load Balancer di livello 4.

- non conoscono il contenuto dei pacchetti
- si basano su:
  - indirizzo IP sorgente e destinazione
  - numero di porta TCP
  - SYN e FIN

Binding table. I pacchetti relativi alla stessa connessione sono inviati allo stesso server. Il Load Balancer mantiene una tabella di corrispondenza connessione TCP e server (binding table). Il Load Balancer usa del bit SYN per identificare nuove connessioni (nuova entry sulla binding table) e del bit FIN per identificarne la fine.

Percorso dei pacchetti. L'architettura (batteria di server + Load Balancer) è chiamata:

- a due vie: se sia i pacchetti in arrivo che quelli in partenza attraversano il LB.
- ad una via: se solo i pacchetti in attivo attraversano l'LB.

La traduzione degli indirizzi. La riscrittura dei pacchetti segue comunemente l'approccio NAT, i pacchetti IP vengono riscritti dal LB sostituendo l'indirizzo VIP con quello del server.È necessario ricalcolare la checksum TCP.

Il meccanismo di traduzione si può basare su IP o su indirizzo MAC.

Si immagini un cluster (batteria di server + Load Balancer) realizzato seguendo l'architettura ad una via:

Meccanismo basato su IP.

- Il Load Balancer prende il pacchetto in ingresso e sostituisce l'IP destinazione da VIP a IP del server scelto.
- Il server genera la risposta e come source address viene specificato VIP (mascheramento dell'IP).

Meccanismo basato su MAC.

- Il Load Balancer prende il pacchetto in ingresso e lo forworda a livello 2 (server e LB sulla stessa rete fisica).
- Tutti i serve memorizzano il VIP sulla loopback e quindi riconoscono il pacchetto come proprio.
- Il server genera la risposta e come source address viene specificato VIP.

Algoritmi di scheduling.

- statici: random, round robin
- dinamici:
  - basati su info del client (assegnazione basata su P del client o della porta),
  - su stato del server (assegnazione basata su numero di connessioni TCP attive, uso attuale di cpu e disco)

Load Balancer di livello 7.

- conoscono il contenuto dei pacchetti: Load Balancer guarda l'header della GET, oppure anche il body di una POST.
- si basano su:
  - contenuto dell'URL
  - cookies
- servono per forza 2 connessioni: una tra il client e il load balancer, l'altra tra il load balancer e il server. Non sarebbe possibile altrimenti per il Load Balancer ispezionare il contenuto della request HTTP.
- Per rendere meccanismo più efficente si possono usare connessioni pronte oppure server pre-allocati.

Algoritmi di scheduling.

- Assegnazione basata su:
  - cache affinity: utilizzo massimo delle cache dei server.
  - richieste HTTP con lo stesso cookie assegnate allo stesso server.
  - prima richiesta di una certa risorsa assegnata al server meno carico, richieste successive della stessa risorsa assegnate allo stesso server.
  - richieste vengono allocate in base al contenuto.

Un cloud provider può mettere a disposizione un numero di server virtuali che può variare nel tempo: il Load Balancer (di livello 4/7), oltre ad eseguire gli algoritmi di scheduling, decide anche, in tempo reale, se aumentare o diminuire il numero di server.

CDN: rete di distribuzione dei contenuti

I fornitori di contenuti distribuiscono risorse a milioni di utenti e hanno bisogno di servire gigabyte di dati al secondo ai loro utenti.

Le CDN (content delivery networks, reti di distribuzione di contenuti) tentano di migliorare le prestazioni del Web per i fornitori di contenuti.

Consegnano i contenuti agli utenti finali da server multipli, geograficamente distribuiti, situati ai margini della rete (web server standard + le cosiddette "repliche"). (Ai margini della rete, ossia vicino agli utenti).

Architettura. Le CDN formano un'enorme rete di overlay al di sopra della dorsale IP e sotto le applicazioni Web e i servizi Web.La CDN, rete di distribuzione dei contenuti, più estesa al mondo è quella Akamai.

Akamai

Ogni fornitore di contenuti che è cliente di una CDN ha bisogno di trasferire ai propri customer contenuti statici e dinamici. Per motivi di scalabilità non è possibile realizzare una rete separata per ciascun cliente, quindi viene realizzato un unico servizio di distribuzione globale.

Il servizio di distribuzione di Akamai è basato su cluster distribuiti situati presso i PoP di vari ISP oppure direttamente PoP Akamai.

Akamai standardizza le esigenze dei fornitori di contenuti gestendo contenuti sia statici che dinamici. I contenuti statici sono forniti da server Web standard di Akamai.I contenuti dinamici necessitano di un server specializzato, copia del server del fornitore (replica).

L'idea generale è questa:

- I fornitori di contenuti hanno i loro server originali.
- Diventano clients di Akamai per scalare.
- Akamai è proprietaria di alcuni PoP, oppure mette i suoi server nei PoP di altri ISP (conviene anche ai provider, perché hanno server in casa senza dover mandare traffico verso un upstream)
- L'architettura di Akamai (batterie di web server standard + repliche + link ad alta velocità + forte monitoraggio dello stato della rete) permette di offrire un servizio di qualità ai client.
- I contenuti statici vengono memorizzati dentro i server di Akamai in modo permanente.
- I contenuti dinamici vengono trasferiti dai server originali alle repliche di Akamai (assemblati sulle repliche Akamai con un linguaggio di markup apposito).
- Una volta arrivati sulle repliche ed assemblati, i contenuti vengono distribuiti ai consumatori.

Come funziona una CDN.

- I server Akamai memorizzano in modo permanente solo i contenuti statici dei clienti, invece i contenuti dinamici vengono prelevati dal sito originale del fornitore.
- Per trasferire i contenuti all'interno della rete vengono utilizzati percorsi Internet ad alta velocità.
- Tutti i tipi di contenuti dinamici e statici possono essere memorizzati nella cache per il tempo specificato da un attributo TTL (poi vanno prelevati di nuovo dal sito originale del fornitore).
- Inoltre i contenuti dinamici e statici vengono assemblati in volo (nell'infrastruttura della CDN) e non sul server del fornitore.

Il sistema di misurazione della rete di Akamai **monitora lo stato** e le prestazioni della rete in diversi punti critici. I valori raccolti di latenza e larghezza di banda vengono utilizzati per:

- selezionare repliche grazie a informazioni ottenute dal monitoraggio.
- determinare percorsi internet ad alta velocità.
- ottimizzare i parametri TCP, ad esempio la finestra di congestione tcp in base al prodotto banda latenza.

I percorsi internet ad alta velocità. I percorsi internet ad alta velocità sono utilizzati per trasferire in modo efficiente contenuti dinamici dalla fonte remota (sito del forntitore) alle repliche locali (server che consegnano i contenuti agli utenti finali). Se si utilizza internet standard si ha latenza maggiore e maggiore packet loss. Se si utilizza il cammino Akamai la latenza sarà minore così come il packet loss. Com'è possibile? Akamai ha dei PoP molto performanti e li sfrutta per raggiungere la destinazione.In più si possono usare i tunnel etc...

Assemblaggio dei contenuti. Come già è stato menzionato i contenuti dinamici e statici vengono assemblati in volo nell'infrastruttura della CDN. Si migliorano le prestazioni assemblando frammenti dinamici e contenuti statici nelle repliche. Akamai ha creato un linguaggio di markup chiamato ESI utilizzato per per l'assemblaggio di componenti delle pagine Web:

- il client remoto fa una hit al server replica e riceve una normale pagina Web.
- i componenti (pezzi) della pagina Web sono stati incollati insieme dal server replica (non nel sito del fornitore). Ogni componente ha il proprio TTL.
- In questo modo è possibile sfruttare in modo migliore le cache.

Meccanismi di distribuzione globale: DNS redirection.

Il customer che risolve con il DNS il nome del sito originale del fornitore, come fa a usufruire dei servizi della CDN? Tutte (o alcune) delle richieste fatte al DNS vengono reindirizzate al server CDN. Il CDN Akamai utilizza il reindirizzamento DNS per fornire contenuti parziali o completi.

Quando si distribuisce il contenuto completo del sito:

- tutte le richieste DNS per il server di origine vengono reindirizzate al server CDN.
- il server di origine è nascosto a tutto il mondo tranne che al CDN.

Quando si distribuiscono contenuti parziali del sito:

- la prima hit è al server d'origine (del fornitore).
- il sito di origine modifica alcuni URL ≪embedded≫ in modo che solo quelli vengano reindirizzati al CDN.
- la richiesta DNS viene reindirizzata ad Akamai (Akamai utilizza una gerarchia di server DNS parallela).
- DNS Akamai traduce la richiesta nell'indirizzo IP di una replica Akamai (edge server, posizionata vicino al client).

URL rewiting. Utilizzata per la distribuzione parziale dei contenuti.

- Il server di origine reindirizza i client verso la CDN riscrivendo opportunamente i link all'interno delle pagine Web on the flight
- Per automatizzare questo processo, le CDN forniscono speciali script che analizzano in modo trasparente il contenuto delle pagine Web e sostituiscono gli URL ≪embedded≫.

Esempio di DNS redirection e URL rewiting.

Quindi cosa succede quando un utente tramite browser richiede la risoluzione di un nome di dominio per recuperare contenuti da un cliente Akamai? Cosa ci aspettiamo succeda:

(1) un browser richiede la traduzione di un nome al DNS per recuperare contenuti da un fornitore (cliente Akamai).
(2) il DNS server autorità del fornitore restituisce un record contenente un CNAME (alias) contenente un nome di dominio della rete Akamai.
(3) Il DNS di default del client vede che è stato utilizzato un CNAME e che il nome ricercato dal client corrisponde ad un nome nella rete Akamai.
(4) Il DNS di default del client reindirizza le ricerche verso il nuovo dominio. Una gerarchia di server DNS Akamai risponde alla richiesta di traduzione del nome, restituendo i migliori due server edge Akamai. I server vengono scelti in base a:
    - l'indirizzo IP del server DNS locale
    - il nome del cliente Akamai
    - il nome del contenuto richiesto

La tecnica fondamentale per indirizzare le richieste dei client verso un particolare server CDN è il DNS redirection.

Vengono usati gli alias (un alias specificato nel campo CNAME) dai fornitori per ridirezionare le richieste di risoluzione dei nomi su name server autorità non propri, ma della CDN.

Esempio 1.

Es. dig www.trenitalia.com
www.trenitalia.com IN CNAME trenitalia.com.edgekey.net.
trenitalia.com.edgekey.net IN A 104.106.125.73
~~~~~~

dig trenitalia.com.edgekey.net

trenitalia.com.edgekey.net. 21233 IN CNAME e17136.a.**akamaiedge**.net.
e17136.a.akamaiedge.net. 1 IN A 104.106.125.73
~~~~~~

Discussione. È saltato fuori **akamaiedge** cercando di risolvere trenitalia, perché è avvenuta una name resolution redirection. Nel mondo CDN la gerarchia dei DNS è parallela a quella a cui sono abituati gli utenti in internet standard.

Esempio 2.

Esempio di distribuzione dei contenuti parziale nella risoluzione di www.adobe.com .

(1) Il client chiede di risolvere www.adobe.com al suo name server di default.
(2) Viene rintracciato il name server autorità di adobe che restituisce l'IP del web server di adobe.
(3) Il client fa un HTTP request al web server di adobe.
(4) Adobe risponde con una pagina. Dentro la pagina c'è un'oggetto ≪embedded≫ che corrisponde all'URL www.images.adobe.com?. Gli URL sono "akamaizzati" Ossia l'URL potrebbe essere già modificato in modo visibile, altrimenti cambia on the flight.
(5) Il name server di default chiede al name server autorità di adobe di risolvere il nome, ma questa volta gli viene restituito un CNAME.
(6) Il name server di default prosegue la risoluzione sulla gerarchia di name server di Akamai.
(7) Verrà restituito l'IP del server replica da cui prelevare le immagini.

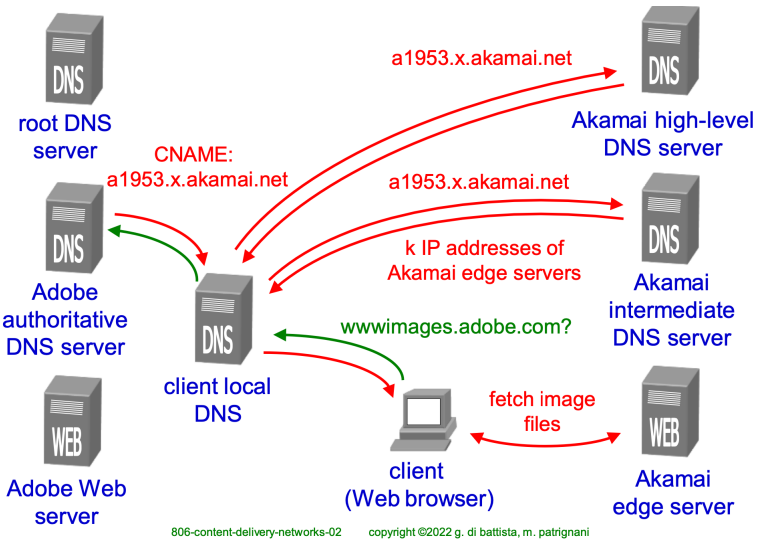

FIGURE 9. a sinistra tutta la gerarchia name server di internet standard, a destra la gerarchia name servere della CDN. Le frecce sono relative alla risoluzione di www.images.adobe.com?.

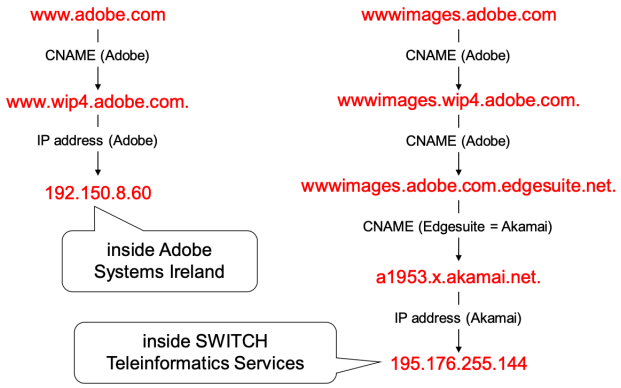

FIGURE 10. distribuzione dei contenuti parziale: a sinistra adobe gestisce la distribuzione delle pagine, a destra akamai gestisce le immagini. I contenutisono distribuiti dall'Irlanda e da un punto Akamai situato da chissà dove.

Osservazione. Non è detto che sia Akamai ad assemblare i contenuti dinamici. Un fornitore per mantenersi indipendente e voler cambiare tecnologia facilmente potrebbe non voler dipendere in modo così vincolato ad una CDN.

Cache DNS. Alcuni di questi step potrebbero essere bypassati grazie all'utilizzo di cache. Come al solito viene manipolato il TTL in modo da avere il giusto controllo sulla schedulazione.

Il caching è uno strumento potentissimo per aumentare l'efficienza, tuttavia può ridurre la capacità di una CDN di indirizzare i client verso i server ottimali.

Per garantire che i clienti siano aggiornati sul server appropriato da utilizzare, i server DNS di Akamai (di basso livello, non root) impostano TTL relativamente piccoli.

Diversi tipi di CDN

CDN single-ISP. Il CDN si comporta come un singolo Internet Service Provider con un AS specifico, con una rete globale che raggiunge una buona copertura geografica. Ad esempio Akamai AS20940:

- Stabilisce peerings BGP con gli AS della rete eyeball
  - presso gli IXP.
  - utilizzando i peerings PNI.

La rete eyeball è un termine usato dagli ingegneri che si riferisce a una rete di accesso i cui utenti principali usano la rete per "guardare le cose" (navigare in Internet, leggere la posta elettronica, ecc.) e consumare contenuti.

CDN multi-ISP. il CDN (ad esempio, Akamai, di nuovo) distribuisce i server nel maggior numero possibile di punti di presenza (POP) degli ISP globali:

- server collocati nella rete eyeball, per servire i suoi clienti diretti ed il cono clienti degli ASes.
- server collocati in una rete di fornitori di transito.

peering (commerciali) CDN. Una CDN collabora con altre CDN per la distribuzione dei contenuti.Le CDN peered forniscono prestazioni migliori a un costo relativamente basso rispetto alle CDN solitarie. Ad esempio in Russia per portare i contenuti serve che una CDN europea si accordi con quella russa.

TCP: IMPLEMENTAZIONE

L'implementazione di TCP è ottenuta dalla composizione di più algoritmi:

Slow start: crescita esponenziale.

- Quando si instaura una connessione, la finestra di congestione cwnd è inizializzata alla dimensione del più grande segment utilizzabile.
- Ogni segment riscontrato prima del timeout fa aumentare la finestra di congestione di un mss.
- Se un segment viene perso (scade timeout) allora cwnd viene riportata al valore iniziale.
- La conseguenza di questo meccanismo è che se la trasmissione procede senza perdita di segment la finestra cwnd cresce esponenzialmente nel tempo.

Si introduce un parametro soglia la **ssthresh.**

- Se cwnd $\leq$ ssthresh allora la finestra cwnd aumenta di un mss per ogni ack (slow start).
- Se cwnd $\geq$ ssthresh cwnd aumenta di $\frac{mss^2}{cwnd}$ (congestion avoidance).

Congestion avoidance: crescita lineare. Per avere crescita lineare, ossia la finestra cresce globalmente di un mss con n pacchetti ricontrati, cwnd' = cwnd + $\frac{mss^2}{cwnd}$.

n pacchetti che posso spedire* coefficiente che fa crescere la finestra = 1.

In formule si traduce così:

$\frac{cwnd}{mss} * \frac{mss^2}{cwnd} = 1$ mss

La retta che descrive la crescita lineare ?

- Se un segment viene perso:
  - Se la causa è la scadenza di timeout o se il segment perso non è uno, ma una grande quantità (non viene riscontrato nessun pacchetto della catena) allora **ssthresh è portata alla metà del minimo tra la finestra di controllo di flusso e la cwnd** (ma deve valere almeno 2mss) .
    - ssthresh = $\frac{min(fwnd,cwnd)}{2}$, con ssthresh $\geq$ 2mss.
    - cwnd è portata a 1 mss.
  - Se vengono riscontrati i pacchetti successivi si usa la tecnica del fast recovery fast retransmit.
    - ssthresh = $\frac{min(fwnd,cwnd)}{2}$, con ssthresh $\geq$ 2mss.
    - cwnd è portata al valore di ssthresh.

IL PIANO AIMD

Il comportamento di due connessioni TCP può essere studiato sul piano aimd. Piano che sull'asse delle x ha il throughput di c1 (quanti pacchetti al secondo spedisce c1).

Sull'asse delle y ha il throughput di c2 (quanti pacchetti al secondo spedisce c2).

Un punto sul piano indica la banda contesa tra c1 e c2 su un link continentale in un certo istante.

SCENARIO IDEALE: TCP HA COMPLETA CONOSCENZA DELLO STATO DELLA RETE

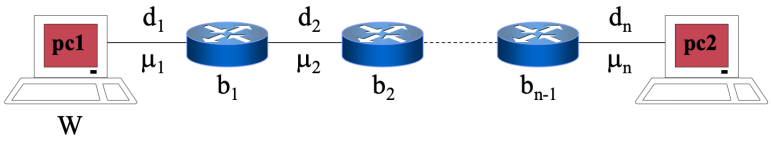

FIGURE 11. pc1 e pc2 sono coinvolti in una connessione TCP. pc1 è il mittente, pc2 il destinatario

Si assume che TCP conosca:

- il valore di banda $\boldsymbol{\mu}$ espressa in $\frac{bit}{s}$ di ogni link
- la latenza **d** (delay) espressa in ms di attraversamento di ogni link
- un buffer dei router **b** espresso in bit
- la finestra di congestione **W**
- un pacchetto è composto da γ bits

Osservazioni:

- si assume che non ci sia traffico nella rete, così tutta la banda può essere usata per la connessione.

- si assume che pc1 abbia dei dati di dimensione grande da inviare a pc2 (tutti i pacchetti inviati hanno dimensione massima)
- pc2 non invia dati e la sua finestra di controllo del flusso è arbitrariamente alta.
- gli ack contengono 0 bits

In stato stazionario (a regime) pc1 non dovrebbe inviare più di $\boldsymbol{\mu}_*$=min($\boldsymbol{\mu}_i$) bit/sec, altrimenti il buffer di qualche router crescerebbe all'infinito. Nel transitorio è possibile che il buffer di un router si riempia, ma a regime no.

Di solito è rispettata la relazione $\boldsymbol{\mu}_1 > \boldsymbol{\mu}_*$. Ossia la scheda di rete di un pc è molto veloce, sarebbe possibile inviare più $\frac{bit}{s}$ di quelli che consente lo stato della rete $\boldsymbol{\mu}_1 > \boldsymbol{\mu}_* = \min(\boldsymbol{\mu}_i)$. Si suppone quindi che il link più lento (che ha minore banda) si trovi nel percorso da pc1 a pc2.

Si può intuire facilmente che non si può ridurre la velocità della propria scheda di rete. Per non congestionare la rete quindi pc1 invierà dei bit, alla sua velocità $\boldsymbol{\mu}_1$, poi rimarrà in attesa, e poi ritrasmetterà. pc1 può γ mandare un pacchetto ogni $\frac{\gamma}{\mu_*}$. Si ottiene l'occupazione minima dei buffer del router se i pacchetti vengono mandati equidistanti.

Esempio scemissimo: $\min(\mu) = 10\ \frac{bit}{s}$
$\gamma = 5$ bit
banda max: $\mu = 20\ \frac{bit}{s}$
numero di pacchetti al secondo : 2 pacchetti
un pacchetto ogni quanti secondi : $\frac{1}{2}$ s
Quanti pacchetti in coda? Al massimo uno, quello che sta passando in quel momento.

Il tempo per ricevere un ack di un pacchetto, ossia il tempo di roundtrip, **rtt** (facendo riferimento alla figura sopra) è pari alla somma di tutte le latenze + tempo di immissione dei pacchetti sui link: $2d_1 + ... + 2d_n + \frac{\gamma}{\mu_1} + ... \_ \frac{\gamma}{\mu_n}$.

Piano di consegna ottimale. Spedendo pacchetti ogni $\frac{\gamma}{\mu_*}$. ed equidistaziati. In questo modo non si accumula nessun pacchetto utilizzando la banda maggiore possibile $\boldsymbol{\mu}_*$.

Self-clocking. TCP deve inviare un pacchetto ogni $\frac{\gamma}{\mu_*}$. secondi, e il tempo viene scandito grazie ad un meccanismo di auto-sincronizzazione. (il sistema operativo non deve preoccuparsi di timer né nulla del genere).

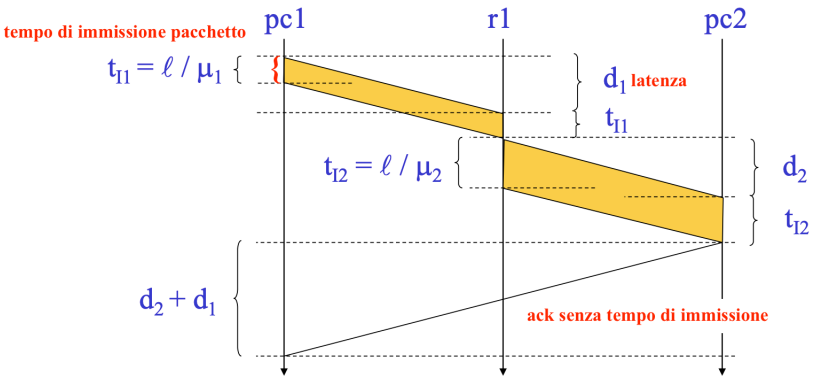

FIGURE 12. pc1 invia un pacchetto in un tempo $\frac{\gamma}{\mu_1}$. Arriva al router r1 dopo un tempo d1. r1 invia il pacchetto in un tempo $\frac{\gamma}{\mu_2}$ ed arriva su pc2 dopo un tempo d2. Infine parte l'ack da pc2 che arriva a pc1 dopo un tempo d1+d2.

PRODOTTO BANDA LATENZA

Qual è il valore ottimale per la finestra di congestione cwnd (W nell'esempio sopra)?

Selezionare un valore per W significa decidere quanti pacchetti (bit) possono rimanere nella pipe/ in flight senza essere unacknowledged.

$\frac{W}{\gamma}$ = numero di pacchetti che posso spedire quando la finestra di congestione è pari a W.
$\frac{\gamma}{\mu_2}$ = tempo immissione della banda più piccola.
Se $\frac{W}{\gamma} * \frac{\gamma}{\mu_2} <$ rtt si sottoutilizza la rete.
Se $\frac{W}{\gamma} * \frac{\gamma}{\mu_2} >$ rtt si rischia di sovraccaricare le code dei router.
Allora $\frac{W}{\gamma} * \frac{\gamma}{\mu_2}$ = rtt e quindi finestra di congestione = latenza complessiva * banda più piccola $\mathbf{W = rtt * \mu_2}$

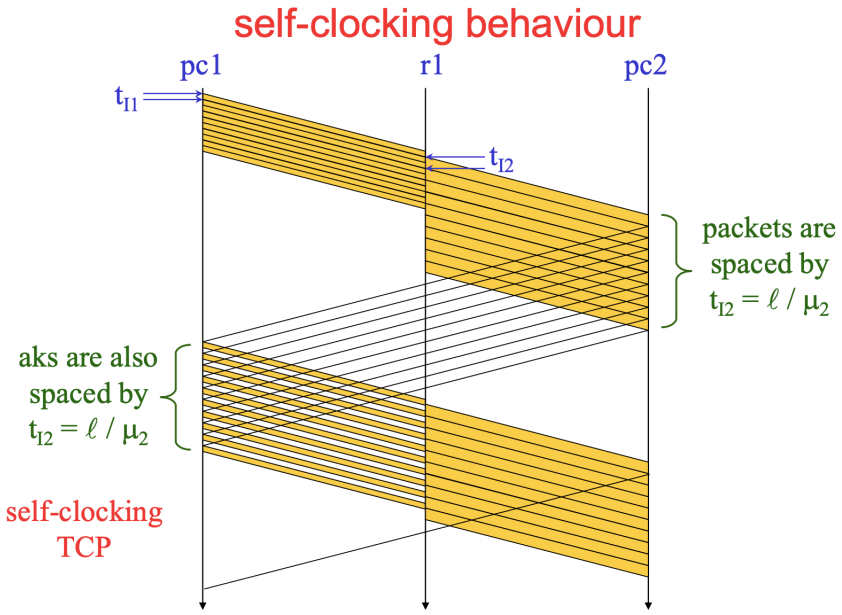

FIGURE 13. pc1 invia 10 pacchetti consecutivi, poi si ferma, a causa della dimensione della finestra W. Nel transitorio (burst iniziale) pc1 invia a r1 un pacchetto ogni $\frac{\gamma}{\mu_1}$. La coda di r1 è sollecitata dal burst e si riempie. Quando pc1 riceve il primo ack si riapre la finestra e può inviare un pacchetto. Visto che pc1 non invia pacchetti finché non riscontra gli ack, a regime pc1 trasmetterà i suoi pacchetti alla velocità dell'hop attraversato più lentamente (il collo di bottiglia). In questo caso pc1 invia un pacchetto ogni $\frac{\gamma}{\mu_2}$

Nella vita reale (e quindi non nel TCP ideale) questo prodotto viene cercato empiricamente facendo slow start congestion avoidance .

Infatti in una rete stabile vi è la necessità di stimare (rapidamente) come impostare la finestra di congestione sul prodotto banda-ritardo senza avere conoscenze sulla rete. L'unica informazione che usa TCP è se c'è o meno packet loss.

TCP ideale	TCP reale
W = rtt * μ_2	W = slow start, congestion avoidance
rtt = $2d_1$+...+ $2d_n$+ $\frac{\gamma}{\mu_1}$+..._ $\frac{\gamma}{\mu_n}$.	rtt = α rtt + (1-α) M

TABLE 2. TCP ideale e reale a confronto

Osservazione: Se rtt molto grande la crescita lineare è lentissima. Per questo si usa slow start. La discussione continua sulle slides...

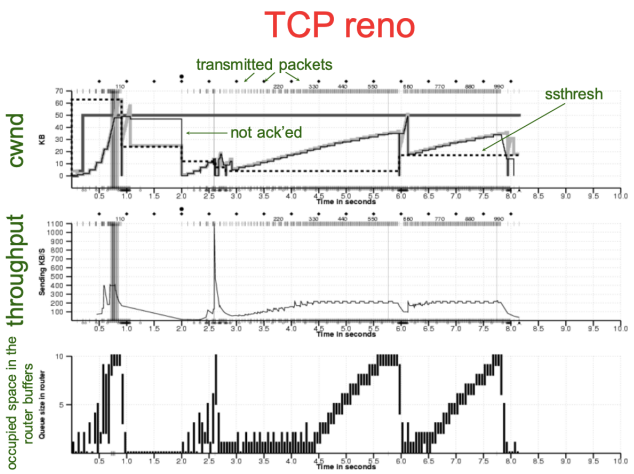

FIGURE 14. TCP reno: implementazione di TCP che ha andamento a dente di sega e rispetta tutte le considerazioni fatte fino ad ora.

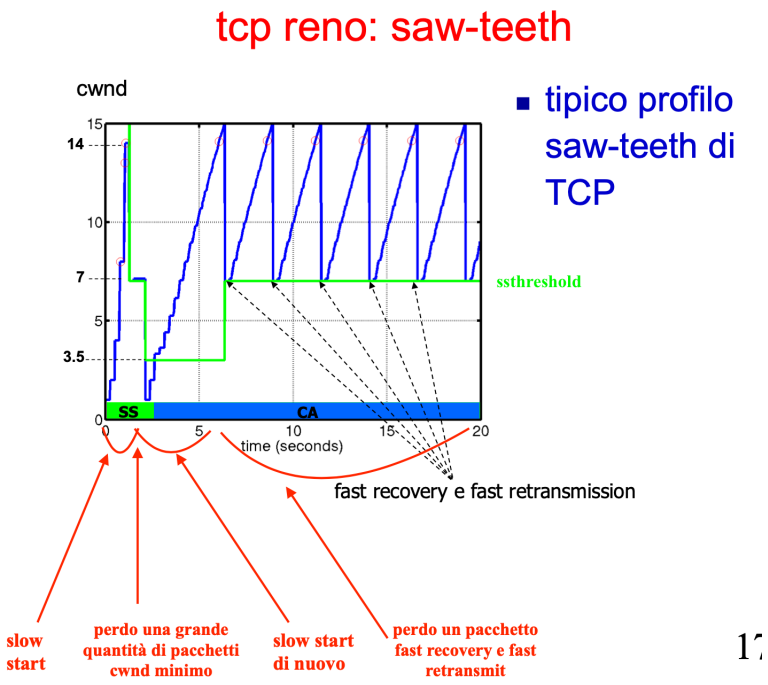

FIGURE 15. TCP reno

Pv6

Gli indirizzi IPv4 sono a 32 bit, gli IPv6 a 128 bit. Un indirizzo IPv6 è quattro volte più grande rispetto ad un indirizzo IPv4. Tuttavia l'header "minimo" IPv6 è solo 2 volte più grande l'header minimo IPv4. Questo perché si è scelto di semplificare l'header IP, facilitando in questo modo il carico computazionale dei router e limitare l'utilizzo di banda. L'header IPv4 dispone dei seguenti campi:

- version
- **header length :** eliminato in IPv6. L'header IPv6 ha dimensione fissa.
- type of service
- total length
- **id, flags, fragment offset:** eliminati in IPv6. Sono tutti campi relativi alla frammentazione.
- ttl
- protocol
- **checksum:** eliminata in IPv6. Per alleggerire il valoro dei router che ad ogni hop devono ricalcolare la checksum a causa del decremento del ttl. (La checksum funziona in questo modo: viene confrontato il valore della **checksum** ricevuto con quello ricalcolato dal destinatario. In questo modo si verifica la presenza di errori).
- Options

L'header IPv6 (40 byte totali senza nessun extension header) dispone dei seguenti campi:

- version: 0110
- traffic class: come TOS in IPv4 permette di specificare priorità dei pacchetti.
- flow label: permette di riconoscere pacchetti appartenenti allo stesso flusso anche se le porte TCP/UDP fossero cifrate. Due pacchetti appartengono allo stesso flusso se hanno stesso IP mittente stesso IP destinatario e stesso valore della flow label.
- next header: è possibile specificare degli extension header che vengono visti dal pacchetto IPv6 come protocolli intermedi. Qui vengono implementate tutte le options IPv4. Il vantaggio di usare il meccanismo degli extension header (instead of the options field) è quello della scalabiltà rispetto al numero di opzioni che si possono usare, e rispetto al carico assegnato ai router. Infatti per i router è più semplice gestire un header di lunghezza fissa rispetto ad uno di lunghezza variabile.
- payload length: la dimensione del payload. Oltre 64 KB si parla di Jumbo payload.
- hop limit: come ttl rinominato.

Next header nell'ordine prestabilito dalla daisy chain. I vari tipi di next header sono:

Hop by hop options. Le informazioni contenute dentro questo header vanno processate da tutti i nodi. Utile per allertare tutti i router oppure avvisare che si tratta di un Jumbo frame

Destination. Le informazioni contenute dentro questo header vanno processate da tutti i nodi stabiliti nell'header **routing.**

Routing. Come source routing in IPv4 permette di specificare la lista di router che si vogliono attraversare

Fragment. L'host può decidere di frammentare i pacchetti (i router intermedi non possono farlo). I link che supportano IPv6 devono accettare almeno 1280 byte, l'host prima di inviare un pacchetto mette in atto un meccanismo di MTU path discovery. Se non esegue MTU path discovery è costretto ad inviare pacchetti di dimensione minima.

Header relativi alla sicurezza. IPv6 confronta nativamente IPsec. Garantisce autenticazione e confidenzialità. Sai che il mittente è autentico e sai che il destinatrio è autorizzato.

Destination: Le informazioni contenute dentro questo header vanno processate dal nodo destinazione.

Tipologie di indirizzi. In IPv6 gli indirizzi possono essere:

- unicast: identificano un nodo
- multicast: identificano un gruppo di nodi, se un host vuole mandare un pacchetto a un indirizzo multicast la connessione sarà del tipo uno a molti.
- anycast: identificano un gruppo di nodi, se un host vuole mandare un pacchetto a un indirizzo multicast la connessione sarà del tipo uno a uno.

Gli indirizzi Unicast.

- unspecified: :: con un indirizzo unspecified si rappresenta la 0/0, viene usato per la prima richiesta al DHCP, viene usato nel meccanismo di DAD
- loopback: ::1 come in IPv4
- IPv4 compatible e mapped: anche se hanno un formato leggermente diverso, si tratta di indirizzi IPv6 che contengono indirizzi IPv4. Servono per la transizione tra IPv6 e IPv4.
- link local: validi all'interno di uno scope, tutte le interfacce di un host si assegnano automaticamente un indirizzo link local. Si tratta di un concetto più forte rispetto agli indirizzi privati, però esattamente come quest'ultimi sono locali e non possono essere annunciati in internet. Il formato è interessante. In IPv6:
  - il prefisso è di 64 bit.
  - l'host id è di 64 bit. Questo valore può essere ottenuto in vari modi. Può essere generato randomicamente, configurato manualmente, oppure può essere calcolato utilizzando lo standard EUI 64. Lo standard prevede che l'indirizzo MAC venga tagliato in due (24 bit + 24 bit) e in mezzo vengano aggiunti 16 bit. Se si vuole costruire un indirizzo globale il numero ottenuto va complementato. La diffusione di indirizzi da cui è possibile risalire al MAC address non è una scelta ottima in termini di sicurezza e di privacy.
  - Un indirizzo link local si ottiene concatenando un prefisso all'interface id ottenuta tramite il formato EUI 64.
- unique local: validi all'interno di un'organizzazione sono "quasi univoci" e quindi favoriscono di molto la possibilità di "fusione" tra enti. Il formato prevede che venga generato un numero random che rappresenta il global ID dell'organizzazione
- global: come gli IP pubblici in IPv4. La gerarchia degli indirizzi global è la seguente:
  - IANA
  - RIPE etc...
  - ISP (possiedono /32)
  - organizzazioni (possiedono /48)
  - reti private (possiedono /64)

Gli indirizzi Multicast.

- sono indirizzi scoped
- il formato prevede un group ID.. Alcuni esempi di indirizzi multicast:
  - gli indirizzi del gruppo all-nodes multicast: l'indirizzo FF01::1 identifica tutte le interfacce sullo stesso nodo, FF02::1 tutte le interfacce sullo stesso link, FF05::1 tutte le interfacce della stessa organizzazione, FF0E::1 tutte le interfacce su internet.
  - gli indirizzi solicited-node per ogni indirizzo unicast/anycasta\ assegnato.

Gli indirizzi Anycast.

Gli indirizzi di un host. Ogni host IPv6 ha questi indirizzi:

- loopback.
- n link local per n interfacce.
- indirizzo del gruppo all-nodes multicast.
- indirizzo solicited-node multicast.
- tutti gli altri indirizzi unicast, anycast, multicast assegnati.

ICMPv6

Il protocollo ICMPv6 condensa ICMPv4, ARP e IGMP, e quindi oltre alle funzioni che aveva in IPv4 (controllo, segnalazione errori), ha due nuove funzionalità:

- neighbor discovery (ARP)
- gestione dei gruppi multicast (IGMP)

Ma prima di tutto il formato del pacchetto ICMPv6: ICMPv6 viaggia dentro IP, ha un header e un payload. Tra i diversi campi c'è la checksum, che però nel calcolo non considera il campo *hop limit.*

I messaggi di errore comunicati da ICMPv6 sono host unreachable, packet too big, time excedeed.

ICMPv6 segnala errori: MTU path discovery. Il messaggio packet too big viene utilizzato nel processo di MTU path discovery. Il contesto è il seguente: un host deve capire se frammentare i pacchetti al liv. 3 oppure no. L'host conosce l'MTU del proprio link ed invia un pacchetto di dimensione pari alla sua MTU alla destinazione che vuole raggiungere.

(1) Il pacchetto arriva a destinazione.
(2) Oppure torna indietro un messaggio ICMPv6 "packet too big" inviato da un router incontrato lungo la strada. In questo caso l'host invia un pacchetto di dimensione pari alla nuova MTU scoperta e reitera il procedimento finché il pacchetto non arriva a destinazione con successo.

Per cercare di aumentare le dimensioni dell'MTU ogni tanto l'host può inviare dei pacchetti sonda di dimensione più grande di quanto è consentito dall'MTU minima incontrata nel percorso.

ICMPv6 fa redirect. come ICMPv4 redirect, ma viene utilizzato indirizzo link local e viene controllato che l'hop limit sia 255 (non si uscito dalla rete locale).

ICMPv6 manda neighbor solicitation e neighbor advertisement. I vantaggi di usare ICMPv6 al posto di ARP sono vari:

- può utilizzare meccanismi di cifratura previsti da IPsec e quindi eliminare problemi di sicurezza introdotti da ARP (es. attacchi ARP spoofing).
- può ridurre in maniera molto significativa il traffico inviato. (Non viene utilizzato traffico broadcast).

La conseguenza è quindi quella di un protocollo più efficiente e sicuro rispetto ad ARP.

Come funziona ARP:

- conosco IP ma non conosco MAC address.
- invio pacchetto di LIV. 2 con indirizzo MAC mittente (il mio) e come destinazione indirizzo BCAST.

In IPv6 si decide di affidarsi a un protocollo di livello 3 e non 2.

Premessa: **ad ogni indirizzo IPv6 unicast corrisponde un indirizzo multicast solicited-node** (n indirizzi unicast n multicast solicited-node), formato aggiungendo gli ultimi 24 bit dell'indirizzo al prefisso ff02::1:ff00:0/104. Tutte le interfacce che hanno lo stesso indirizzo solicited-node (ossia hanno gli ultimi 24 bit dell'indirizzo MAC uguali) sono iscritte allo stesso gruppo multicast.

Il contesto è lo stesso:

- conosco IP ma non conosco MAC address.
- calcolo l'indirizzo multicast solicited-node corrispondente all'indirizzo IPv6 che voglio raggiungere.
- invio un pacchetto **multicast ICMPv6 di neighbor solicitation** che:
  - al LIV. 3 ha come indirizzo IP mittente il mio, come destinazione l'indirizzo multicast solicted-node.
  - al LIV. 2 come indirizzo MAC mittente il mio, come indirizzo destinazione un indirizzo MAC multicast mappato a quello solicited-node.
  - payload: l'indirizzo IPv6 del destinatario.

MAC mittente: il mio MAC address	MAC destinazione: indirizzo MAC multicast
IP mittente: il mio IP	IP destinazione: IP multicast solicted-node

TABLE 3. pacchetto di neighbor solicitation

- il destinatario, se presente, risponde con un pacchetto **unicast** di **neighbor advertisement.**
- l'indirizzo fisico del destinatario viene memorizzato nella mia neighbor cache.

ICMPv6 fa nud. Ogni nodo tiene traccia dello stato di mutua raggiungibilità con i nodi vicini. Se un nodo non ha informazioni sulla raggiungibilità di un vicino:

(1) **invia pacchetti unicast di neighbor solicitation**
(2) se non ottiene risposta, cancella la entry relativa a quel vicino nella cache e ripete il procedimento di neighbor solicitation (inviando pacchetti multicast)
(3) se questo fallisce, il vicino è irraggiungibile.

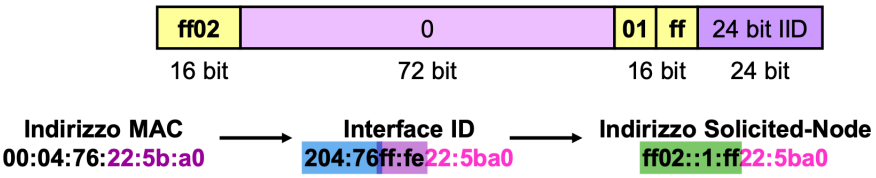

FIGURE 16. Costruzione di un indirizzo solicited-node. Un host ha un indirizzo link local che rispetta il formato EUI 64. Si è ottenuto l'host id di un indirizzo link local in questo modo: il MAC address è stato diviso in due (24 bit + 24 bit), sono stati aggiunti 16 bit (ff:fe), è stato complemetato. Essendo link local un indirizzo unicast, viene anche costruito un indirizzo multicast solicited-node. Vengono presi gli ultimi 24 bit dell'indirizzo e viene aggiunto il prefisso ff02::1:ff.

ICMPv6 manda router solicitation e router advertisement. Il meccanismo di **autoconfigurazione stateless** permette ai nodi IPv6 di connettersi alla rete senza dover configurare manualmente gli indirizzi, senza dover utilizzare un server DHCP.

Quando una macchina si sveglia in una rete IPv6:

- avrà un indirizzo link local per ogni interfaccia, configurato autonomamente. In IPv6 una connessione point-to-point non richiede configurazione.
- vorrebbe avere altri indirizzi (magari globali) basati sugli Interface id. Tuttavia mancano i prefissi.

Come fanno gli host ad apprendere i prefissi.

(1) I router inviano periodicamente i router advertisement a tutti i nodi (mandano pacchetti all'indirizzo multicast FF02::1).

(1) All'avvio, l'host configura gli indirizzi link-local.Genera un indirizzo link-local per ciascuna interfaccia. Con duplicate address detection verifica che l'indirizzo sia unico sul link.Se è unico, lo assegna all'interfaccia.
(2) Utilizzando gli indirizzi link-local, il nodo può già comunicare. Invia un pacchetto di router solicitation a tutti i router della LAN (manda pacchetto all'indirizzo multicast FF02::2).
   Ogni router risponde con un pacchetto unicast di router advertisement.
   Per ogni router advertisement che riceve:
   - aggiunge il router alla lista dei router disponibili.
   - configura un indirizzo per ogni prefisso nell'annuncio.
   - rimane in ascolto dei messaggi di router advertisement.

Router advertisement. I router advertisement sono pacchetti che contengono informazioni utili per gli host:

- specificano l'indirizzo IPv6 link local del router.
- specificano l' indirizzo di livello 2 del router.
- indicano se il router. è disponibile ad essere il default router.
- possono fornire il valore di alcuni parametri come hop limit, MTU.
- **contengono una lista di prefissi assegnati al link.**

Gli host utilizzano l'indirizzo link-local del router come default gateway, e ottengono indirizzi IPv6 concatenando il loro interface id ai prefissi.

Tempo di vita degli indirizzi. Nel router advertisement ad ogni prefisso sono associati due tempi di vita in secondi:

- preferred lifetime. Se vale 0 l'indirizzo è "deprecato" e non può essere utilizzato per nuove connessioni, ma può essere usato per quelle ancora attive.
- valid lifetime. Se vale 0 l'indirizzo non è valido e non può rimanere assegnato all'interfaccia, viene rimosso.

Host renumbering.

(1) basta configurare un nuovo prefisso sul router e porre a zero il preferred lifetime di quello vecchio.
(2) alla fine del processo di renumbering il vecchio prefisso viene rimosso dal router.

Router renumbering. Se cambia il prefisso associato ad un'organizzazione, devono essere riconfigurati i prefissi sui router. Il protocollo di router renumbering permette di cambiare indirizzi e prefissi sui router da una stazione di gestione. Tuttavia non affronta il problema del cambiamento dei record DNS.

Riconfigura i router da remoto attraverso messaggi di controllo originati da una stazione di gestione. Prevede obbligatoriamente l'uso di IPSEC per autenticare i messaggi.

Autoconfigurazione stateful.

- configurazione interamente manuale
- DHCPv6

ICMPv6 fa DAD. Prima di assegnare un indirizzo ad una interfaccia, il nodo verifica se l'indirizzo è già assegnato ad un altro nodo sullo stesso link.Non dovrebbe mai accadere se si usa l'autoconfigurazione stateless.Il nodo invia una serie di pacchetti di neighbor solicitation:

- l'indirizzo sorgente è l'unspecified address.
- l'indirizzo destinazione è l'indirizzo solicited-node corrispondente a quello che si vuole assegnare.

Se non riceve in risposta un neighbor advertisement, l'indirizzo è valido e può essere assegnato.

IPv6 e multihoming

L'architettura Shim6 consente il multihoming IPv6 senza compromettere la scalabilità del sistema di routing globale (non crescono esponenzialmente le tabelle dei router), utilizzando indirizzi aggregabili dal provider.

I provider annunciano solo le meno specifiche. Il bilanciamento di carico, così come la fault redundancy vengono affidati a Shim6. I router non si accorgono dell'esistenza di questo protocollo.

L'idea su cui si basa shim6: l'indirizzo IP condensa due idee semanticamente diverse. Un IP permette di identificare un nodo in termini di:

(1) **Id**. Identifica il nodo ai protocolli di strato superiore.
(2) **Posizione.**Identifica la posizione del nodo nella topologia di rete.

Un host mette in piedi una connessione TCP. L'IP che si sta utilizzando per la connesione TCP si rivela essere poco appropriato.

Normalemnte BGP ricalcolerebbe la best e cambierebbe percorso, utilizzerebbe il link primary o backup a seconda dell'esigenza, farebbe bilanciamento di carico. Tutto questo senza interrompere la connessione TCP.

È possibile utilizzare l'extension header di shim6 per rispettare questi stessi obiettivi.

Da un certo momento in poi al LIV. 3 i pacchetti che mando dentro la rete cambiano indirizzo, ma quando arrivano a destinazione cambio di nuovo e lo riporto a quello usato all'inizio della connessione. Liv 4. non se ne accorge grazie a shim6, il "prestiggiatore".

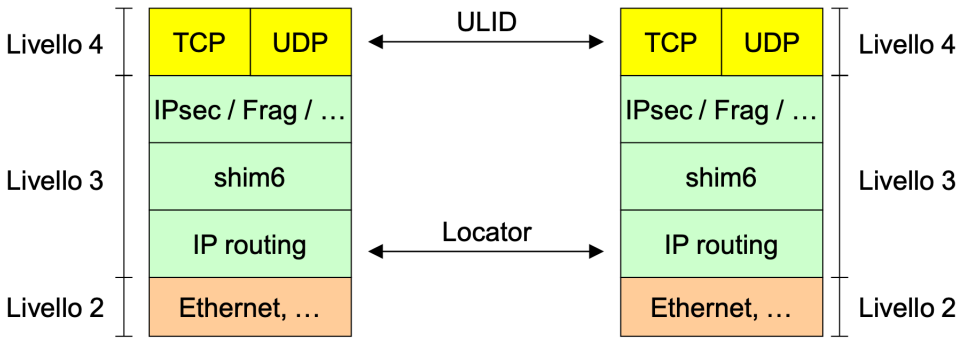

Figure 17. due endpoint che usano shim6.

- Quindi il livello 4 vedrà sempre gli stessi indirizzi iniziali che identificano gli EndPoint, (ULID) quindi la connessione TCP non verrà interrotta.
- Il livello 3 vedrà gli indirizzi correnti (Locator).

Shim6 di sx e Shim6 di dx si scambiano dei tag "la connessione si basa su questo indirizzo IP, ma io possiedo anche questi altri, potrebbero arrivarti pacchetti da altri indirizzi, sono sempre i miei"

Multihoming non fatto solo con BGP, ma anche con shim6: dentro un customer i prefissi di vari provider, il customer può usarli tutti.

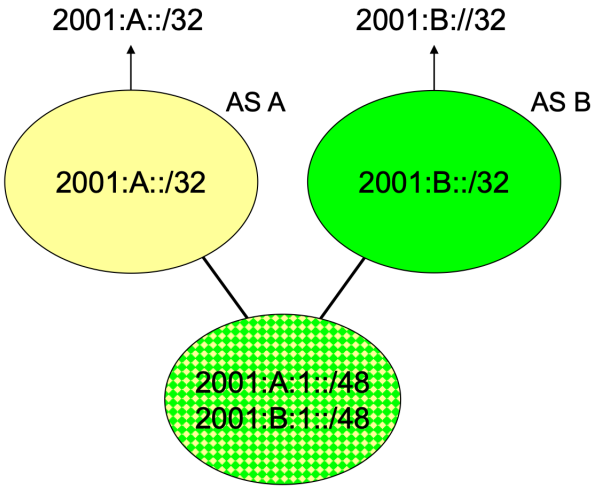

FIGURE 18. scenario AS che usano IPv6

(1) Due provider annunciano la 2001:A::/32 e la 2001:B::/32 e danno al proprio customer le: 2001:A::/48 2001:B::/48
(2) Un host per iniziare una connessione usa il suo IP pubblico 2001:A::1/48.
(3) Con BGP il pacchetto arriva fino a destinazione.
(4) Non voglio annunciare le /48, ma voglio fare bilanciamento di carico. Allora che faccio?
(5) Gli IP ULID rimangono sempre gli stessi, (es. 2001:A::1/48), ma se una linea è più congestionata di un'altra, uso distinti IP locators (ad esempio 2001:B::1/48).
(6) Solo gli host si accorgono della traduzione i router e le tabelle di instradamento BGP no.

TRANSIZIONE IPV4 IPV6

- meccanismi basati su dual stack. Si tratta di meccanismi di compatibilità, non di transizione. Un esempio è happy eyeballs. In un host dual stack:
  - si cerca di stabilire una connessione IPv6 e poi, solo se non ci riesce, una IPv4 (l'utente rischia di attendere troppo tempo).
  - le connessioni IPv4 e IPv6 si avviano assieme, con un lieve anticipo per IPv6.
- meccanismi basati su traduttori di protocollo: SIIT, NAT 64

SIIT: stateless. Sei in una rete IPv6 (rete del cliente) e devi raggiungere soltanto IPv4 (rete del server). In mezzo tra client e server rete IPv6 .

(1) dopo la query al DNS (conosco address IPv4 del server), query al DHCP server per avere IPv4 mittente temporaneo.
(2) prendo pacchetto con IPv6 mittente translated, IPv6 destinazione IPv4mapped (del server) e lo instrado verso il traduttore SIIT.

IP mittente: IPv6 translated	IP destinazione: IPv6 IPv4mapped

TABLE 4. pacchetto instradato verso il traduttore SIIT

(1) L'indirizzo IPv4 mittente/destinatario viene tirato fuori (decapsulato) e sparato verso il server. La traduzione effettuata dal traduttore è stateless perché gli indirizzi sono desunti direttamente dagli indirizzi IPv6.

(1) Al ritorno il traduttore fa la conversione opposta.

IP mittente: IPv4 temporaneo	IP destinazione: IPv4 server

TABLE 5. pacchetto in uscita dal traduttore SIIT

Vantaggi. semplice, scalabile, robusto. Il traduttore si limita a buttare/ aggiungere pezzetto di indirizzo.

Svantaggi. Richiede la presenza di un meccanismo (DHCP?) per l'assegnazione dinamica degli indirizzi temporanei.Difficile gestire la distribuzione in subnet degli indirizzi IPv4.

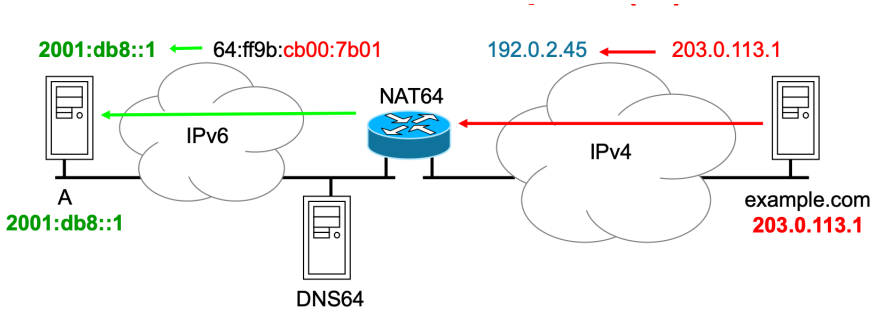

FIGURE 19. scenario NAT64

NAT64: stateful.

(1) macchina A fa query al DNS64 (un name server apposito) e riceve indirizzo IPv6 taroccato (non viene restituito l'address IPv4 del server ma il suo corrispondente IPv6).
(2) pacchetto in ingresso al traduttore avrà come mittente 2001:db8::1 e come destinatario 64:ff9b:cb00:7b01 (due indirizzi IPv6).

IP mittente: IPv6 della macchina A	IP destinazione: IPv6 taroccato

TABLE 6. pacchetto instradato verso il traduttore NAT64

(1) Il traduttore assegna all'indirizzo IPv6 mittente un indirizzo IPv4 attingendo dal pool (da usare sempre come mittente). Mantiene traccia dell'associazione in una tabella di stato.Pacchetto in uscita dal traduttore avrà come mittente 192.0.2.45 (attingendo dal pool) e come destinatario 203.0.113.1.

IP mittente: IPv4 del pool di indirizzi	IP destinazione: IPv4 del server

TABLE 7. pacchetto instradato verso il traduttore NAT64

(1) al ritorno il pacchetto viaggerà su una /24 annunciata da NAT64 (ad esempio 192.0.2.45 che assocerà un indirizzo IPv6).

La macchina A ha la percezione di essere connesso all'host IPv6, il Web server ha la percezione di essere connesso all'host IPv4.

Vantaggi. trasparente ai nodi che lo utilizzano (con SIIT la macchina doveva fare una query al DHCP per farsi assegnare un IPv4 temporaneo che corrispondeva ad un IPv6 translated), si può accoppiare con PAT, usando un solo indirizzo IPv4 per molti client.

Svantaggi. fragile perché è stateful, non permette connettività diretta da estremo a estremo.