Riassunto Impianti di elaborazione

by diverse fonti

Si ringraziano:
Adrinano - Akira - Cheruba - Palla - Zazzu

Tassonomia dei Web Server Scalabili

Progettazione di Servizi Basati sul Web con Esigenze Stringenti in Termini di Prestazioni – Architetture, Modelli e Algoritmi

Gli obiettivi che si propone questa parte sono l'identificazione dei problemi a cui si va incontro nell'ambito dell'analisi delle prestazioni di un web server, la classificazione dei Web-Server scalabili, la distinzione tra architetture localmente distribuite e architetture distribuite, la classificazione e comparazione degli algoritmi usati e l'identificazione delle alternative di progettazione.

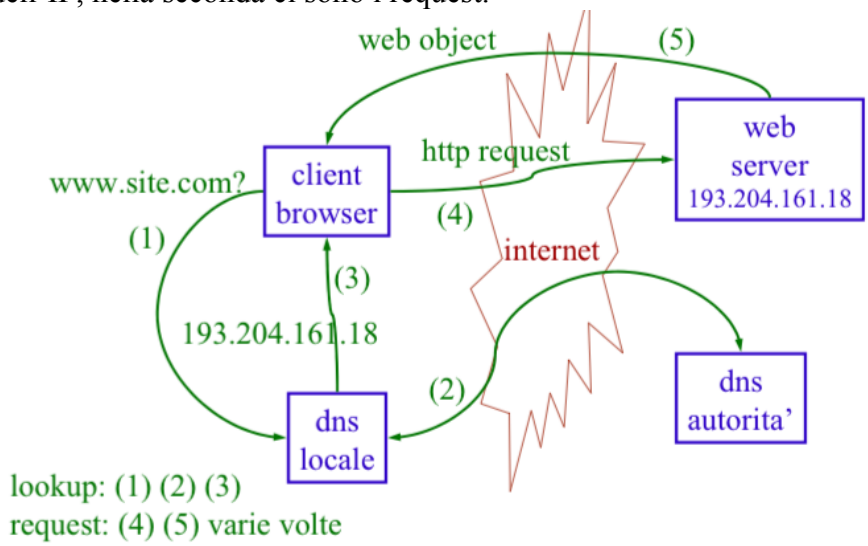
Diamo alcune definizioni preliminari:
Hit: richiesta di un singolo oggetto al server da parte del client (ad esempio un'immagine o un testo);
Page Request: richiesta di una "pagina" composta di solito da vari hit;
Session: sequenza di page request consecutive da parte dello stesso client.
Inoltre, ciascun oggetto può essere statico, dinamico, o sicuro.

Perché ci concentriamo sulle prestazioni dei web server? Si calcoli che siti come google o lycos hanno più di 50 milioni di hit al giorno, e solo nell'edizione 1999 di Wimbledon sono stati fatti quasi un miliardo di hit in soli 14 giorni sul sito apposito, con una giornata di picco con 125 milioni di hit. Inoltre il web della nuova generazione è ben diverso dalla prima, dove non giravano tante informazioni "critiche" e la maggior parte delle pagine era formata da testo. Al giorno d'oggi un'organizzazione si valuta in base al proprio sito, le informazioni sul web sono spesso di livello critico e crescono gli elementi dinamici. C'è insomma necessità di **QoS** (Quality Of Service).
In particolare, quando si parla di Web Service, si hanno due tipi di QoS: la QoNS e la QoWS (rispettivamente Quality of Network Service e Quality of Web Service). La prima è relativa alla rete messa magari a disposizione di un Provider, e si misura in tempi di latenza, numero di pacchetti scartati, disponibilità del servizio.
La Qualità di un Web Service invece si misura in questo modo: si sceglie un servizio, si sceglie una metrica con la quale misurare il servizio (es. disponibilità, efficienza, sicurezza, accessibilità: alcune misure sono relative al servizio, altre al sistema, altre ad entrambi, inoltre alcune metriche sono su percentuale, altre sono binarie) e si sceglie una soglia, dopodiché si conta in che misura la metrica del servizio supera tale soglia.

Il Service Level Agreement (SLA) è appunto l' "accordo sul livello del servizio". In pratica le specifiche non funzionali che il servizio deve soddisfare.
Esempi: il sistema funziona per X% del tempo o X% delle richieste ha un tempo di risposta di Y secondi o meno.

Chi offre SLA sono più i Network Provider che le organizzazioni, perché i primi hanno pieno controllo della propria dorsale, mentre gli ultimi controllano le componenti dell'architettura solo in parte.

Vediamo cosa succede in una semplice richiesta al server. La macchina client chiede l'accesso a un sito www.site.com e quindi dialoga con il DNS locale per farsi dire l'IP corrispondente al nome specificato. Il DNS locale dialoga (eventualmente) con il DNS autorità per farsi dire tale indirizzo, e poi lo riporta alla macchina client. Quest'ultima, finalmente in possesso dell'indirizzo IP, instaura la connessione con il server ed effettua vari http request seguiti ciascuno da un oggetto restituito. In pratica nella prima fase c'è un lookup dell'IP, nella seconda ci sono i request.

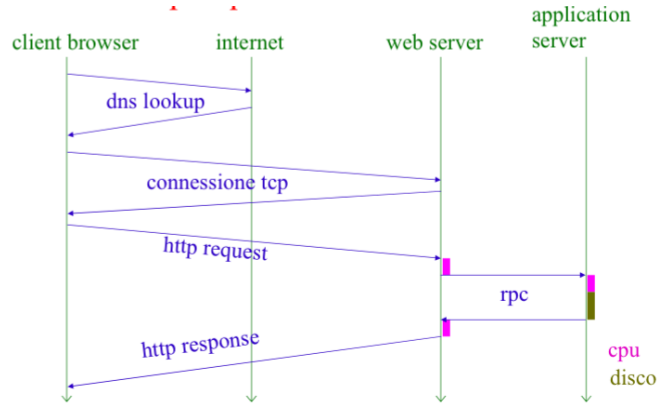


I possibili problemi che possono verificarsi sono a livello di dns, web server, internet e proxy. Nel primo caso potrebbero esserci indirizzi ip non più validi in cache e il dns potrebbe andare in timeout, nel secondo caso potrebbero esserci problemi di sovraccarico o non raggiungibilità, nel terzo caso sovraccarico dei link o router e nell'ultimo caso problemi di oggetti non più validi.
Le soluzioni possono essere a livello di rete, di sistema (overprovisioning di cpu, memoria o numero di server con distribuzione locale o geografica) o di infrastruttura (dns e cache).

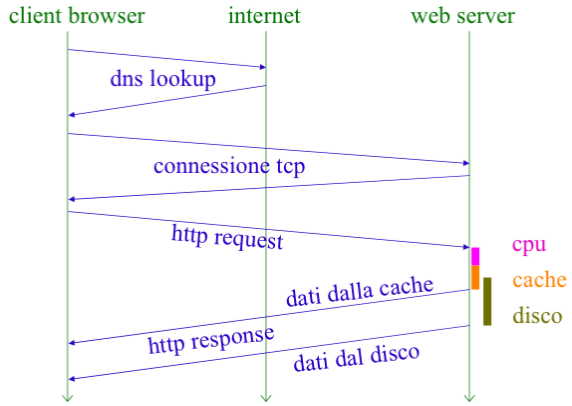
I requisiti dei **sistemi basati su più server** comprendono accesso rapido, trasparenza, scalabilità, robustezza, disponibilità e affidabilità.

Nel caso statico dopo il dns lookup avviene prima la connessione tcp col server trovato e poi le richieste http, a cui il web server risponde prima con i dati della cache e poi quelli del disco. Nel caso dinamico all'http request segue una comunicazione con l'application server tramite *rpc*.

Dinamico



Statico



Nella progettazione di un servizio bisogna analizzare in dettaglio le caratteristiche del **carico sul servizio**, secondo modelli qualitativi (solo la media) e quantitativi (termini di probabilità).
Gli aspetti da considerare sono:
- i tempi delle richieste;
- dimensione, interesse e tipo degli oggetti richiesti;
- tipo di servizio (statico, dinamico o sicuro).

In genere i tempi di interarrivo hanno distribuzione poissoniana, cioè è più probabile che in un certo intervallo di tempo T ci siano un certo numero di arrivi piuttosto che ce ne siano di meno o di più (ad esempio è difficile che nell'arco di un minuto, ci siano meno di una decina di arrivi ed è difficile che ce ne siano mille! Il numero di arrivi più probabile è la cima della "campana" di poisson).
Invece la distribuzione della durata delle sessioni e anche del think-time dell'utente tra una richiesta e l'altra è cosiddetta *heavy tailed*, ovvero valori alti sono presenti con probabilità non trascurabili (difatti è possibile che un utente resti sul sito durante una stessa sessione anche per molto tempo come un'ora, o due ore così come cinque minuti, e lo stesso vale per il tempo che ci mette a fare nuove richieste).
Per quanto riguarda la dimensione degli oggetti richiesti, la distribuzione è ancora heavy tailed, ma molti oggetti sono piccoli. Per l'interesse, gli oggetti seguono la legge di Zipf: frequenza x rango = costante, ovvero più è importante un oggetto (rango basso), più sono frequenti gli accessi a quell'oggetto (in realtà in questo caso sembra molto banale questa legge). Per quanto riguarda il tipo, spesso gli oggetti più acceduti sono html e immagini.
Per quanto riguarda le caratteristiche del tipo di servizio, basta fare considerazioni logiche:

oggetti statici prevedono meno lavoro di quelli dinamici in termini di cpu e disco. Oggetti sicuri prevedono ancora più lavoro in termini di cpu e di rete, ma non necessariamente di disco.

Classificazioni di Riferimento

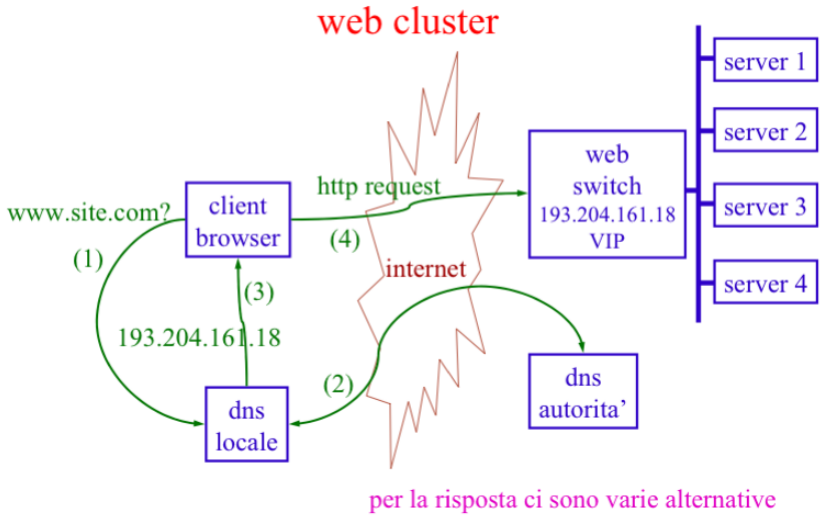
Un web server "scalabile" è basato su vari server distribuiti. Ogni volta che arriva una richiesta, viene fatto uno scheduling e scelto il server migliore. Lo scheduling prevede tre cose: un meccanismo, un algoritmo e una entità.
L'**algoritmo** stabilisce quale sia il server migliore;
il **meccanismo** è il modo in cui viene assegnata la richiesta al server scelto dall'algoritmo;
l'**entità** è l'oggetto fisico che esegue l'algoritmo e il relativo meccanismo.
Gli algoritmi di scheduling si dividono in statici e dinamici. I primi si dicono così perché lavorano senza informazioni (potrebbero scegliere un server a caso, oppure ragionare in maniera round-robin, oppure far scegliere all'utente, tipo mirror), i secondi hanno informazioni sul client e/o sullo stato dei server.
Ecco una prima overview della tassonomia dei meccanismi.

Meccanismo	Entità	Livello
Name Resolution (locale/ globale)	dns	sessione
Http Redirection (locale/ globale)	web server	page request
Packet Redirection (locale)	web switch	page request / hit / sessione

In particolare si distinguono meccanismi applicabili a distribuzioni di server locali e globali (geografiche).

Distribuzione Locale – Web Cluster

Un'architettura web cluster è caratterizzata dal fatto che tutti i server si trovano nella stessa locazione (ad esempio un Internet Data Center – IDC), e sono raggiungibili tramite un unico indirizzo IP (detto VIP: Virtual IP) mostrato all'esterno, mentre i singoli server hanno indirizzi IP privati, che possono essere assegnati a diversi livelli della pila iso-osi. Quindi il dns locale riceve dall'autorità un unico indirizzo ip (il VIP) da riferire al client, con cui poi instaura la connessione TCP. Tale connessione è instaurata con un **web switch** che si occupa dello scheduling (algoritmo e meccanismo) e comprende richieste e risposte. La richiesta dunque è fatta al web switch, per la risposta invece ci sono varie alternative divise in "a-due-vie" (la risposta la ridà lo stesso switch) e "a-una-via" (la risposta la dà direttamente poi il server scelto).



Cos'è di preciso il web switch? E' il componente che smista le richieste, effettuando il mapping tra VIP e indirizzo dei server. Può essere un hardware "special purpose" (fatto apposta) oppure un modulo software di un SO. Il web switch effettua un controllo fine sulle richieste e può farlo a livello di hit, page request o sessione.

I web switch di livello 4 non conoscono il contenuto delle pagine (content-blind) e si basano su indirizzo ip di sorgente e destinazione, numero di porta tcp, e i bit syn e fin dei pacchetti tcp che indicano l'apertura e la chiusura di connessioni (ovvero lavorano a livello di trasporto/rete). I web switch di livello 7 sono content-aware e sono basati su contenuto dell'url, cookies e ssl id (valore usato per connessioni sicure).

Quindi, l'attuale tassonomia riscontrata è composta da quattro tipologie, che sono tutte le combinazioni possibili tra web switch di livello 4 o 7 con architettura ad una o due vie. All'interno di queste quattro tipologie, sono possibili vari algoritmi di scheduling, statici o dinamici, e vari meccanismi.

Gli switch di livello 4 operano dunque a livello di tcp/ip. I pacchetti relativi ad una stessa connessione (identificati tramite il syn) sono fatti corrispondere ad uno stesso server (tramite una tabella connessione-server mantenuta dallo switch). Nell'architettura a due vie lo switch "riscrive" anche i pacchetti in partenza. Tale riscrittura segue l'approccio NAT (Network Address Translating). Significa che il VIP comune gestito dallo switch viene tradotto nell'IP della rete interna dei server e viceversa per i pacchetti in uscita. Questo significa però dover ricalcolare le checksum tcp. Nell'architettura ad una via i pacchetti in uscita non passano per lo web switch, e i server modificano i pacchetti in uscita ponendo come mittente l'indirizzo VIP. Un'alternativa dell'architettura ad una via è evitare la riscrittura a livello IP e usare gli indirizzi MAC per il reindirizzamento ai server. Gli algoritmi possibili sono statici e dinamici. Tra gli statici ci sono random (assegnazione casuale: nessuna informazione sullo stato dei cluster né sulla storia delle assegnazioni) e round-robin (assegnazione "a turno", unica informazione: l'assegnazione precedente). Tra gli algoritmi dinamici annoveriamo la partizione dei client (assegnazione effettuata in base all'indirizzo dei client, in questo modo è possibile controllare la Quality of Web Service per gruppi di utenti), assegnazione in base al carico dei server (osservato dallo switch in termini di connessioni attive, appreso dal server

stesso tramite informazioni su uso di cpu e disco o ricavato tramite emulazioni di richieste dallo switch ai server).
Esempi di architetture web switch di livello 4 a due vie sono *cisco local director*, *cabletron lsnat*, *alteon web systems*, *linux virtual server*. A una via sono *ibm tcp router & network dispatcher* e *lsmac*.

Gli switch di livello 7 operano invece a livello di applicazione. Ispezionano il contenuto dei pacchetti http per decidere sull'assegnamento, quindi hanno un parser http. Possono dunque indirizzare le richieste su server specializzati in funzione del contenuto. Dato che possono arrivare al pacchetto http solo dopo averlo estrapolato da quello tcp, devono gestire per intero la connessione tcp e occuparsi degli ack. Anche gli switch di livello 7 si classificano in base ai meccanismi usati e all'architettura a due o una via. In particolare stavolta ci sono quattro particolari architetture possibili, due a due vie (tcp gateway e tcp splicing) e due a una via (tcp handoff e tcp connection hop).
Nel due-vie tcp gateway, lo switch si comporta come un proxy (rappresentante) per entrambe le parti e fa da intermediario alla connessione. In questo modo l'overhead è altissimo: lo switch deve mantenere due connessioni tcp e tutta la pila iso-osi è concentrata su di esso. Nel tcp splicing, che ne è una ottimizzazione, dopo aver instaurato le due connessioni tcp, lo switch effettua una traduzione di indirizzi e numeri di sequenza operando esclusivamente a livello 3 (di rete) in modo tale da "incollare" le due connessioni tcp in un'unica connessione a livello 4.
Nell'una-via tcp handoff, la connessione instaurata tra client e switch viene trasformata in una connessione tra client e server, mentre nella variante connection hop la connessione viene trasferita al server con un protocollo apposito. Nella dispensa tecniche-switch-livello-7-02.pdf è illustrato molto bene il funzionamento della conversione dei numeri di sequenza da parte dello switch nel caso di TCP Splicing, e anche il funzionamento di TCP handoff.
Gli algoritmi usati sono molteplici.
- Algoritmi in funzione della sessione: richieste http con lo stesso ssl id o gli stessi cookie sono assegnate allo stesso server, così si evita di identificare più volte lo stesso client;
- Algoritmi in base al contenuto stesso: si possono specializzare i server sul tipo di file (audio, video, html, ecc.) oppure assegnare il server in base alle dimensioni del file (site interval task assignment with equal load – sita-e) per privilegiare i task più leggeri oppure partizionare il file space tra i server con funzioni tipo hash per massimizzare lo hit rate delle cache dei server;
- CAP – Client Aware Policy: classificare le risorse in funzione dell'impatto sulle singole componenti dei server (cpu, dischi, rete), e quindi in base al tipo di richiesta (una query sql avrà impatto sui dischi, un download di un grosso file sulla rete, richieste con alti requisiti di sicurezza sulla cpu). Questa politica è accompagnata da un'assegnazione ciclica delle classi ai server, per diversificarne l'impegno;
- Cache Affinity: tecnica basata su un cache manager a cui è nota la situazione delle cache di tutti i server;
- Locality-Aware Request Distrubution (LARD): la prima richiesta di una certa risorsa è assegnata al server meno carico (con criterio: numero di connessioni). Successive richieste della stessa risorsa sono assegnate allo stesso server, per aumentare la località e quindi lo hit rate delle cache.

Esempi disponibili sul mercato sono: per tcp gateway *ibm network dispatcher*, per tcp splicing *alteon*, *arrowpoint*, e *foundry nets*, per tcp handoff *lard*, per tcp connection hop *resinate central dispatcher*.

Sistemi Web Distribuiti

Qui si parla di distribuzione geografica (server di uno stesso sistema anche molto distanti fra loro). Le architetture possibili sono: siti mirror, server distribuiti, cluster distribuiti. Gli algoritmi di scheduling possono essere a uno, due o tre livelli. Sono possibili vari modelli e metriche.

I **siti mirror** prevedono una stessa informazione replicata su più web distribuiti geograficamente. L'indirizzamento prevede nomi multipli, uno per ciascun sito replicato, e lo scheduling è lasciato all'utente che sceglie il sito da una pagina di partenza. Nonostante sia molto semplice come architettura, la replicazione è visibile, bisogna mantenere la consistenza fra i vari siti e non c'è alcun controllo sulla distribuzione del carico.

Poi ci sono i **server distribuiti**, che sono più server distribuiti geograficamente e replicati, con lo stesso nome ma con indirizzi IP diversi. Quindi è il DNS autorità per il sito (ma non necessariamente) che sceglie il server al primo livello di scheduling. Poi viene previsto un secondo livello di scheduling implementato tramite http redirect. Il client quindi ad ogni richiesta riceve dal dns autorità la coppia indirizzo IP (scelto) + ttl (time to live, ovvero quanto tempo tale indirizzo deve rimanere in cache nei dns intermedi). Gli algoritmi previsti possono essere al solito statici o dinamici.
Lo scheduling dinamico può essere complesso perché funzione del carico e quindi di giorno, ora, fusi orari, distribuzione degli utenti in internet ma anche prossimità tra client e server, e caching nei dns intermedi. A ciò si aggiunge che i tempi di latenza spesso sono indipendenti dalla vicinanza geografica e i link non hanno tutti la stessa capacità.
Le informazioni statiche che si possono ottenere staticamente sono indirizzo IP del client, dal quale si può ricavare la zona di internet (l'Autonomous System) in cui esso risiede, numero di network hop (router tra i due nodi) e numero di AS hop ("zone internet" tra i due nodi).
Le informazioni dinamiche invece sono roundtrip di latenza (tempo che intercorre tra l'invio di un segmento e la ricezione del relativo ack, ottenibile tramite ping), disponibilità di banda sui link e tempo di latenza di una richiesta http (ottenibile tramite un'emulazione). Si noti che per effettuare tali valutazioni serve tempo e traffico addizionale sulla rete. Pare che ci sia un correlazione tra hop (informazione statica) e latenza (informazione dinamica).
Alcune considerazioni sul ttl: si potrebbe porlo uguale a 0 per affidare al dns il pieno controllo dello scheduling, ma non è sicuro che i dns collaborino a questo scopo (perché potrebbero ignorare ttl molto piccoli), inoltre si sovraccaricherebbero troppo, e infine la cache dei browser ignorano comunque il ttl. L'altra alternativa è un ttl dinamico che si adatta al carico attuale dei server e alla frequenza di richieste da un certo dominio. La verità è che gli algoritmi attuali usati dai DNS sono molto complessi (a differenza dei web switch).
Alternative prevedono di non utilizzare i dns bensì redirect fin dal primo livello di scheduling. Oppure utilizzare i dns in primo livello e http redirection (o anche IP tunneling) in secondo livello.

Esperimenti con nslookup. Il comando nslookup permette di trovare quali indirizzi IP restituisce il DNS per un certo nome ad un certo client specificato. Per specificare il server a cui chiedere la risoluzione si digita "server xxx.yyy.zzz", dopodiché si digita il nome del sito (ad esempio www.google.com) e il programma restituisce il nome del sito e i vari alias (come ad esempio www.l.google.com) e gli indirizzi IP restituiti dal server specificato. Quando dice non-authorative answer significa che gli indirizzi IP che sta dando sono cachati in qualche dns intermedio.
Qualche parola sull'http redirect, che è di fatto il secondo livello di scheduling. Esso prevede uno scheduling distribuito (ciascun server può fare un redirect, assegnando la richiesta a qualcun altro). E' completamente trasparente all'utente (ma non al client), e la redirezione può essere verso un nuovo nome o un nuovo indirizzo IP. Le politiche di redirect possono essere molteplici. Bisogna scegliere chi può effettuare redirect (solo il dns o tutti i server), quali pagine ridirigere (tutte, solo quelle che superano una certa soglia in dimensione o numero di hit), e a chi redirigere (round robin, server meno carico, funzione hash, prossimità). L'http redirect è ottimo ma si possono redirigere solo le richieste http e aumenta traffico e tempo di risposta, dato che bisogna mantenere due connessioni http per richiesta).
Sul mercato scheduling a un livello sono *server ncsa*, *cisco distributed director*, *sun scalr* mentre a due livelli c'è *sweb*.

Per ottenere il terzo livello di scheduling basta introdurre anche i web cluster. In pratica sono **web cluster distribuiti** geograficamente. Il nome è unico ma c'è un indirizzo IP (ovvero VIP) per ogni cluster, con indirizzi privati dei server all'interno. I tre livelli di scheduling sono:
- DNS o altro durante il lookup;
- web switch;
- redirect dai web server (o dal web switch se è di livello 7).
Sul mercato a due livelli: *alteon websystems*, *cisco distributed diretcor*, *resonate global dispatcher*, *hydraweb techs*. A tre livelli: *hermes*, *rnd networks*, *radiare wsd-ds*.

Misurazione e Tuning delle Prestazioni di un Servizio Web – Apache JMeter & Apache

Nell'ambito di un servizio web, bisogna poter misurarne e regolarne le prestazioni. La valutazione consente di dimensionare le risorse necessarie all'erogazione di un servizio prima dell'attivazione, la misurazione ne analizza il comportamento in presenza di carico, il tuning ottimizza i parametri di configurazione mentre il servizio è già attivo. Gli strumenti di misurazione sono numerosi, e di vario tipo, sia software che hardware. **Apache jmeter** è una delle soluzioni software, mentre tra gli hardware annoveriamo **spirent avalanche 2500**.
Jmeter, che è uno strumento per testare il carico non solo di applicazioni web-based ma anche per altri ambiti, è scritto in java (per garantire disponibilità di API e portabilità) e consente di configurare dettagliatamente il comportamento degli utenti.

Uso di JMeter.
- Test Plan: insieme di Thread Group, eseguibili contemporaneamente o in sequenza.
- Thread Group: oggetto che simula un gruppo di utenti e genera rapporti sui tempi di risposta del servizio. Un Thread Group ha:
- Numero di Thread: numero di utenti nel gruppo da simulare.
    - Periodo di ramp-up: tempo necessario affinché tutti gli utenti vengano attivati. Come dice la parola stessa (rampa) il numero sale linearmente, quindi si attiva un utente ogni k secondi fino ad arrivare all'ultimo utente attivato all'ultimo istante del periodo di ramp-up.
    - Numero di loop: ciascun thread non si attiva una sola volta, bensì un numero di volte pari a questo valore.

Inoltre un Thread Group può avere al suo interno vari altri sotto-oggetti, come sampler, listener, logic controller, timer, in un ordine che è significativo perché influenza l'ordine di esecuzione delle operazioni (suppongo operazioni di un utente/thread generico). Credo che se un thread group fosse definito senza alcuno di questi sotto-oggetti, non farebbe nulla (e non servirebbe a nulla).

- Sampler: effettua richieste su un server (http, ftp, jdbc, tcp, rpc, ecc.), con diverse proprietà configurabili (ad esempio l'URL della risorsa da richiedere). Lo si può far agire a livello di hit o di richiesta, e gli si può associare un'asserzione (per verificare un valore booleano, come l'esistenza di una stringa nella risposta http).
- Listener: raccoglie dati sulle risposte del server, su file, grafici o tabelle.
- Logic Controller: controlla la sequenza di attivazione dei propri figli. Ad esempio il LC "once only" fa sì che tutti i propri nodi figlio vengano attivati una volta sola (se ad esempio c'è un sampler, farà una sola richiesta) anche se il numero di loop del nodo padre è maggiore di uno. Il "once only" è usato per effettuare login. I figli del Logic Controller possono essere qualsiasi sotto-oggetto che possa far parte di un Thread Group.
- Timer: se per default ogni richiesta avviene subito dopo la precedente, col timer divengono cadenzate. Può servire per introdurre un intervallo tra una richiesta e l'altra, o per cercare di forzare la frequenza delle richieste (tipo fare in modo che arrivino 120 richieste al minuto).

Tutti questi elementi, tranne Sampler e Logic Controller, possono far parte del Test Plan anziché di un Thread Group (esempio: un listener nel test plan può valutare complessivamente i tempi di risposta per tutti i thread group, mentre un timer può controllare in maniera generale la frequenza di richieste da parte di tutti i thread group).

Sulla dispensa impianti-jmeter-04.pdf è ben illustrato il funzionamento di jmeter, con vari esempi. Nel primo esempio si fa un test di carico molto semplice, con un unico Thread Group composto da un unico utente, che fa dieci richieste distanziate 2 secondi l'una dall'altra. (Con un sampler, un timer che agisce sul sampler e un listener che mostra i tempi di risposta). Gli utenti ovviamente si comportano in modo molto più complesso.
Il secondo esempio di test plan verifica i tempi di risposta di un dizionario on-line, effettuando il login e richiedendo ciclicamente un insieme di parole ad intervalli di 4 secondi, verificando se i tempi di risposta siano al di sotto di una soglia, che è di 2 secondi. Con un Logic Controller once-only si fa l'accesso e il login (configurando gli opportuni parametri) e metodo GET o POST a seconda della richiesta. Con un Logic Controller for-each si può, per ogni variabile presente in un array definito dall'utente grazie al programma, mettere tale variabile in un'altra usata dai figli di questo Logic Controller, come la richiesta di una ricerca

di una parola (fra le parole dell'array definito dall'utente). Si mettono anche i relativi timer e un cookie-manager per gestire i login (mi sono registrato a garzantilinguistica per fare lo stesso test, funzionava quasi tutto login compreso, ma non le ricerche delle parole, avrò sbagliato qualcosa nella sintassi sulle variabili lemma).

Ovviamente un test di carico realistico deve stressare notevolmente il server. Per aumentare il realismo, JMeter non utilizza cache, a differenza dei browser. Inoltre per buoni test di carico è opportuno che non girino processi estranei sulle macchine interessate, che le risorse di calcolo siano rivolte esclusivamente al test, che si esegua il test da riga di comando (jmeter lo permette) magari col server grafico completamente disattivato. Inoltre è opportuno riavviare il web server dopo ogni test per rigenerare il pool iniziale di processi e thread (che durante l'esecuzione varia dinamicamente), nonché considerare l'effetto delle cache. E' opportuno utilizzare una buona macchina come client e una macchina non troppo veloce come server.
I passi per un test di carico sono semplici. Dopo aver verificato la conformità alle linee guida, si avvia il web server, lo si setta per il monitoraggio lato server, si avvia il test plan si arresta il monitoraggio lato server e si riavvia il server per ricominciare la nuova iterazione.
Sono definiti alcuni test plan significativi.
basic-u: singolo utente, singolo hit, tempo tra le richieste uniformemente distribuito.
basic-c: singolo utente, singolo hit, tempo tra le richieste costante.
single-user: singolo utente con navigazione completa e tempo fra le richieste uniformemente distribuito.
single-user-r: single-user con tempo di attivazione dei thread uniformemente distribuito.
two-users: due utenti con due percorsi di navigazione e tempo fra le richieste uniformemente distribuito.
Nei test fatti ogni test plan viene istanziato più volte con valori di versi di ramp up e numero di thread. Se i valori sono entrambi troppo alti, il client potrebbe non reggere per le troppe risorse da allocare (non è detto però che essi sovraccarichino il server).

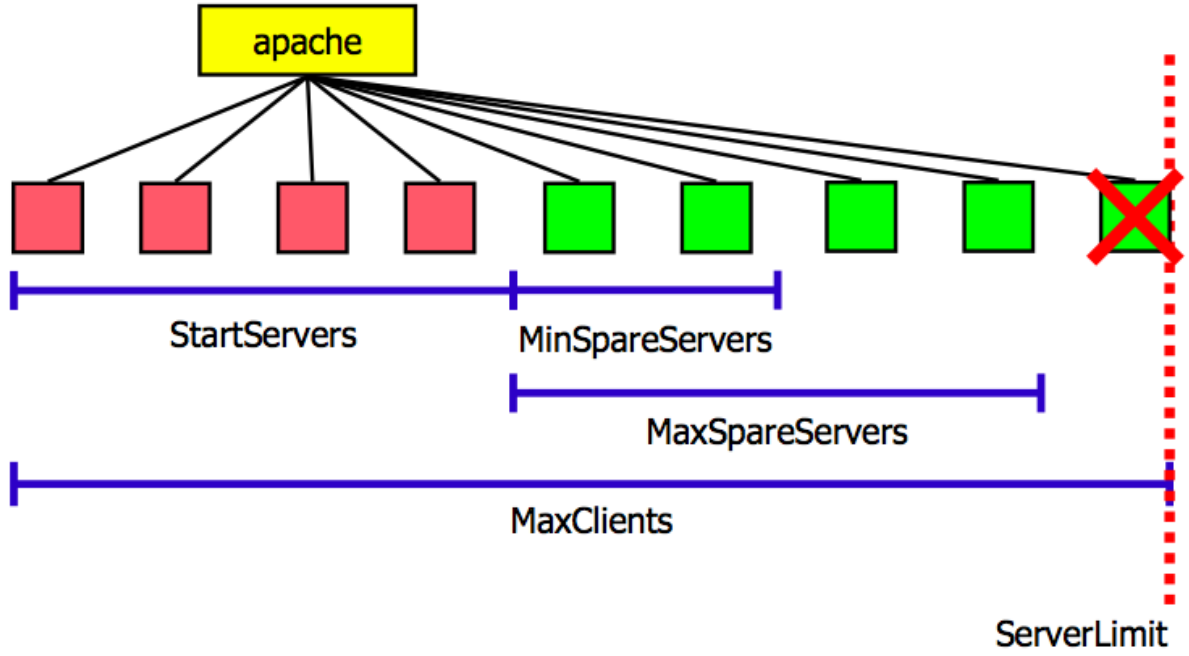
Tuning di un Servizio Web

Nel tuning di un servizio web bisogna controllare i parametri hardware e software per l'ottimizzazione, quindi CPU, RAM, velocità del bus, tempo di accesso al disco, spazio su disco e per il software risorse da pubblicare e numero di risorse impiegate per erogare il servizio (come processi e thread). In Apache molti parametri sono relativi a moduli specifici (i moduli Apache vengono caricati a runtime), altri sono generici.
Tra i parametri generici annoveriamo:
HostNameLookups[off]: risoluzione dell'IP in nome dell'host (on/off/double). Aumenta il tempo necessario a completare una richiesta.
KeepAlive[on]: rende persistenti le connessioni.
KeepAliveTimeout[15]: tempo in secondi in cui un processo figlio deve rimanere in attesa per altre potenziali richieste del client prima di chiudere la connessione. Valori bassi aggiungono overhead di banda e fanno dimenticare lo stato degli algoritmi di slow start e congestion avoidance, valori alti aumentano l'impiego di risorse sul server, facendo sì che i client occupino indefinitamente le socket del server.
MaxKeepAliveRequests[100]: numero massimo di richieste prima che la connessione venga chiusa.

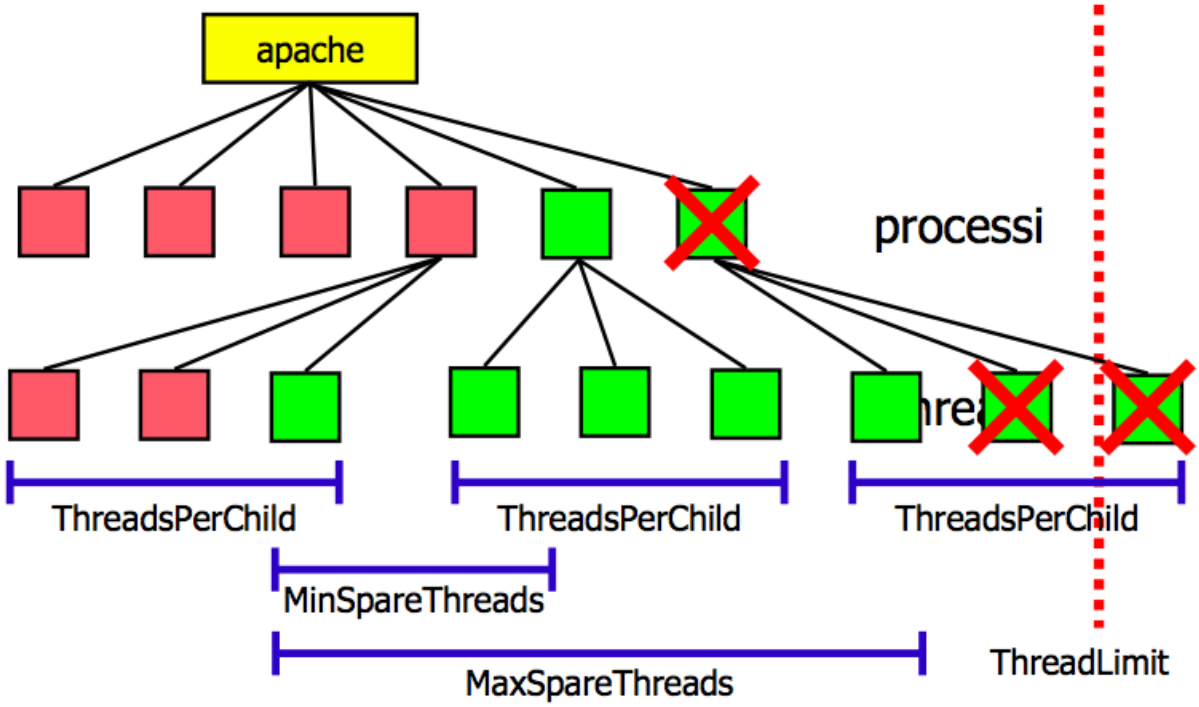
In realtà (citazione sulle dispense) solo l'utente sa veramente quando deve poter concludere la connessione. Sfortunatamente gran parte degli utenti non sa cos'è una connessione e permettere loro di chiudere le connessioni sarebbe senza senso, inoltre un utente che sappia cosa sia una connessione, avrebbe tutti i motivi di chiuderla il più tardi possibile.
In Apache sono importanti per questi scopi i Moduli Multi-Processo (MPM), che gestiscono efficientemente le richieste. Questi sono **Prefork**, **Worker** (a sua volta suddiviso in **Threadpool** e **Leader**) e **Perchild**, ancora in fase di sviluppo.
I parametri di mpm_common sono i seguenti, anche se alcuni non sono utilizzabili in certi mpm:
ListenBackLog[?]: massimo numero di client in attesa di essere serviti (ulteriori richieste vengono ignorate).
MaxMemFree[0]: quantità massima di memoria allocata in KB prima di invocare free().
StartServers[?]: numero di processi figli generati all'avvio del server.
MaxClients[?]: numero massimo di client serviti simultaneamente (ulteriori richieste vengono accodate, se il valore è troppo alto, la gestione causa swapping).
ServerLimit[?]: numero massimo di processi configurabili con MaxClients.
MaxRequestPerChild[10000]: numero massimo di richieste che un processo figlio può servire prima di essere terminato. Limita gli effetti dei memory leak (sprechi di memoria allocata e non rilasciata) e riduce più rapidamente il numero di processi alla riduzione del carico del Server.
StartThread[?]: numero di thread generati all'avvio del server.
MinSpareThreads[?], MaxSpareThreads[?]: numero minimo e massimo di thread inattivi in attesa di servire.
ThreadsPerChild[?]: Numero di Thread generati da ogni processo figlio.
ThreadLimit[?]: numero massimo di thread configurabili con ThreadsPerChild.

Prefork

Ha i due ulteriori parametri:
MinSpareServers[5] e MaxSpareServers[10], che configurano il numero minimo e massimo di processi figli inattivi (in attesa di servire).
In Prefork a servire sono i processi.
Un processo padre lancia StartServers processi figli (che in seguito possono variare).
Il numero di processi figli inattivi viene sempre mantenuto tra MinSpareServers e MaxSpareServers.
MaxClients rappresenta dunque il numero massimo di processi figli.
Funzionamento. Ogni secondo vengono creati N processi, dove N cresce esponenzialmente (quindi +1, +2, +4, +8, fino a 32), finchè non si supera MinSpareServers. Valori di N maggiori di 4 potrebbero essere una potenziale anomalia e vengono riportati nel log. Quando si raggiunge il numero desiderato di processi inattivi, N riparte da 1. Ovviamente non si può superare MaxClients processi.



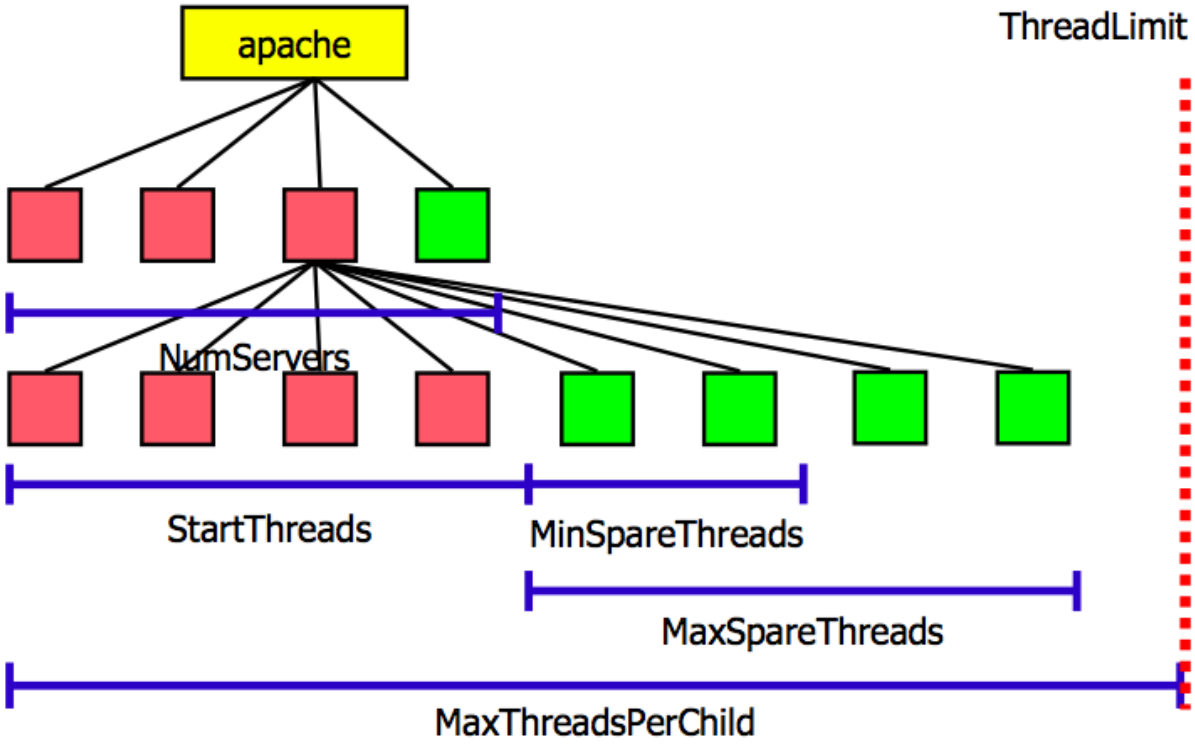
Worker
Lavora con processi e thread (sono loro a servire).
Un processo padre lancia StartServers processi figli (in seguito il numero può variare).
Ciascun figlio lancia ThreadsPerChild thread (più uno di ascolto), il numero è fisso.
Il numero totale di thread viene costantemente mantenuto tra MinSpareThreads e MaxSpareThreads: a tal scopo vengono generati o terminati dinamicamente i processi figli.
MaxClients dunque rappresenta il numero massimo di threads.



Perchild

Lavora con processi e thread.
Un processo padre lancia NumServers processi figli (numero stavolta fissato).
Ciascuno lancia StartThreads threads (numero che stavolta può variare).

Per ogni processo figlio, il numero di thread inattivi viene mantenuto tra min e max generando o terminando thread.
Nuovi parametri: StartThreads (<= ThreadLimit) e NumServers.
Altri parametri generici sono RLimitCPU, RLimitMEM e RLimitNPROC che limitano l'utilizzo delle risorse hardware; LimitRequestBody, LimitRequestFields, LimitRequestFieldSize e LimitRequestLine che limitano le entità delle richieste da parte del client; inoltre ci sono dei moduli sperimentali per gestire il caching o per analizzare lo status.



In conclusione il tuning è un'operazione parecchio complessa ed è difficile fornire una procedura universale. Si noti però che Apache è piuttosto autoconfigurante quindi spesso non c'è bisogno di cambiare i valori di default.

Content Delivery (Distribution) Networks

Una Content Delivery Network (CDN) recapita i contenuti agli utenti finali da vari server sparsi ai confini della rete, migliorando le prestazioni del Web. I cosiddetti "content provider" contrattano con le CDN per l'hosting e la distribuzione dei propri contenuti.

Akamai è la più estesa CDN nel mondo. Molti sistemi importanti fanno affidamento su di essa per servire i tanti Gigabyte per secondo di immagini, animazioni flash e video dei propri siti web.
Misurazioni, algoritmi di selezione dei cammini e di gestione delle cache di Akamai sono private e top secret, però i meccanismi che permettono ad Akamai di ridirigere le richieste dei client sono di dominio pubblico e ben spiegati.

Qualche tecnica di base: dato che ogni CDN dispone di un server in prossimità dei POP (Point of Presence, punti di accesso alla rete) degli ISP, le richieste possono essere inoltrate dinamicamente al duplicato più vicino. Di solito si usa la redirezione DNS o la riscrittura dell'url.
Molti CDN giacciono su sottosistemi per la misurazione delle reti, così possono utilizzare le informazioni dinamiche conosciute per selezionare i server duplicati e determinare i cammini sui quali trasferire i contenuti.
Con la redirezione DNS una request è rediretta ad un DNS autorità posseduto dal CDN, che risolve il nome con IP di uno o più server. La redirezione può essere di contenuti completi (redirezione di tutti i request) o parziali (redirezione soltanto di alcuni URL immersi che vengono modificati dal sito d'origine). Akamai utilizza redirezioni di contenuto parziale.
In particolare Akamai traduce la request di un client che sta richiedendo contenuto nel dominio di un cliente di Akamai, nell'indirizzo IP di un server Akamai vicino, detto Edge Server. Ecco i passi che segue:

- l'utente richiede la traduzione di un nome per ottenere il contenuto di un customer di Akamai;
- il DNS server del customer restituisce un "nome canonico" CNAME come entry, che contiene un nome di un dominio nella rete di Akamai. Questo funge da "alias", permettendo al server DNS di ridirigere le ricerche di nome a un nuovo dominio (quello di Akamai);
- quindi, una gerarchia di server DNS di Akamai risponde alla richiesta di traduzione di nome, determinando e restituendo gli IP degli edge server migliori, in base a IP del DNS locale (o del client, a seconda di dove sia stata originata la richiesta di traduzione), nome del customer Akamai e nome del contenuto richiesto.

Esempio di redirezione DNS di Akamai (traduzione "paro paro" dalla dispensa in inglese)

- Homepage di PCWorld.com
- Un client web emette un request per un oggetto immerso che risiede nel dominio images.pcworld.com; dopodiché richiede al proprio server DNS locale (LDNS) l'indirizzo IP di images.pcworld.com
- Quindi l'LDNS tenta una traduzione di nome per conto del client
    - Quando il name server di PCWorld è contattato per una traduzione di nome, esso comincia la redirezione DNS ritornando una entry CNAME per images.pcworld.com, poiché il contenuto del dominio images.pcworld.com è servito da Akamai
        - Il valore di CNAME in questo caso è images.pcworld.com.edgesuite.net; edgesuite.net è un dominio di proprietà di Akamai
- L'LDNS esegue nuovamente una traduzione di nome, questa volta sul dominio edgesuite.net
    - Sono in seguito eseguite due nuove redirezioni, prima al dominio akam.net (es. adns1.akam.net), poi a a1694.g.akamai.net, dove 1694 è il numero di customer per PCWorld
    - In generale, le redirezioni Akamai includono un numero di customer nel nome del dominio
- Nello stadio finale della traduzione, la rete di Akamai usa una gerarchia di name server Akamai per restituire all'LDNS gli indirizzi IP degli edge server che

dovrebbero garantire un download veloce e devono di solito trovarsi vicini al web client che ha iniziato la request
- L'LDNS viene direzionato al name server akamai.net che comincia il processo di ricerca di un edge server vicino inoltrando l'LDNS ad un DNS server Akamai di alto livello, ad esempio, uno nominato za.akamaitech.net
- Un DNS server Akamai di alto livello è un elemento di un piccolo set globale di server DNS, responsabile di delegare un request DNS ad un appropriato DNS server Akamai di basso livello; generalmente un DNS server Akamai di basso livello è più vicino al LDNS di uno ad alto livello
- Quindi, il DNS server Akamai di basso livello (attualmente chiamato usando il pattern n#g.akamai.net dove # è una cifra tra 0 e 9) ritorna gli indirizzi IP di due edge server che si aspetta offrano alta performance per il client
    - Le macchine che operano come DNS server Akamai di basso livello e come edge server possono essere (e di solito lo sono) le stesse
- Alla fine, l'indirizzo IP dell'edge server viene restituito al web client, che è ignaro di tutte le redirezioni che si sono dovute effettuare

Sulle dispense c'è l'output dei comandi DIG per tracciare tutte le richieste ai vari DNS. DIG è un'utility che serve per sapere in che modo vengono tradotti i nomi di dominio, molto più sofisticata di nslookup ma anche molto chiara (o almeno così dicono…). Le entry dell'output sono: nome richiesto, TTL, tipo di entry, valore restituito. Il tipo di entry (che comincia sempre con IN: internet) può essere A (indirizzo IP), CNAME (canonical name, cioè un alias), NS (name server) o MX (mail exchangaer). Sulle dispense c'è anche l'output del comando whois fatto sull'IP della rete in cui si trovano gli edge server restituiti, per confermare la vicinanza. Si scopre infatti che tale rete è composta dai server Akamai nel backbone di Telecom Italia International. Whois è un tool che mettono a disposizione le organizzazioni come RIPE, e che dato un indirizzo IP dà informazioni sulla locazione della rete di tale indirizzo.
Si noti che a causa della cache è possibile che l'indirizzamento agli edge server migliori sia compromesso. Per questo si usano TTL molto bassi (20 secondi per il DNS di un edge server, 1 ora per le entry che puntano ai DNS Akamai di basso livello, 48 ore per quelli che puntano ai DNS Akamai di alto livello).

Misure su Larga Scala di Akamai
Sono riportate alcune misurazioni sulla CDN Akamai fatte durante un esperimento di 7 giorni in cui ogni 20 secondi 140 nodi di PlanetLab sparsi in tutto il mondo hanno effettuato richieste DNS per uno fra 15 importanti customer di Akamai.
Si nota ad esempio che per Yahoo, rispetto al tempo vengono visti più identificatori IP di edge server di giorno che non di notte.
Rispetto all'ID dell'host PL, la maggior parte degli host vedono tra i 10 e i 50 server, alcuni (i primi) vedono pochi edge server (e sono i più vicini alla rete Akamai), mentre altri vedono quasi tutti gli edge server (e sono quelli più lontani dagli "hot-spots" Akamai).
Si nota anche che Yahoo, Amazon, NY Times ed altre grandi organizzazioni fanno hostare il proprio contenuto a una grande varietà di edge server in molte zone.
(Si nota che la dispensa è piena di grafici non spiegati, per i quali si capisce ben poco – NdD).
(Da qui in poi la dispensa si capisce sempre meno. Quello che segue è un sunto estremamente essenziale che non spiega nulla delle definizioni usate, ad esempio gli approcci multi-ISP e

single-ISP non si capisce affatto cosa significhino, perché non si capisce cosa siano di preciso i server surrogati)
I customer scelgono attraverso un tool chiamato Free Flow Launcher quali contenuti servire per conto proprio e quali far servire da Akamai. Grazie a questo tool, le pagine servite vengono "akamaizzate" (codificando appositamente gli url dei contenuti da far servire agli edge server).

Dietro Akamai – Problemi Generali e tecniche delle CDN

Nella consegna di contenuto parziale il server di origine modifica gli url degli oggetti immersi nella pagina così che essi possano essere risolti dal DNS della CDN.
Nella consegna completa il primo hit arriva al CDN e il server d'origine è nascosto da tutto il mondo tranne che dalla CDN. Tutte le richieste sono servite dalla CDN.
L'approccio *single-ISP* prevede molti server surrogati che possano supportare numerosi oggetti.
L'approccio *multi-ISP* prevede numerosi server surrogati in altrettanti point of presence degli ISP.

Alcune CDN utilizzano la riscrittura degli url anziché lo schema basato su DNS. Il server di origine ridirige i client su diversi server surrogati riscrivendo i link delle pagine generate dinamicamente. Per automatizzare il processo le CDN forniscono script speciali che parsano in modo trasparente il contenuto della pagina rimpiazzando gli url immersi. La tecnica viene anche chiamata "modifica del contenuto".

Ci sono anche le CDN in peering tra di loro, in cui ciascuna consegna i contenuti per conto di ogni altra. Un content provider contratta solo con una CDN, ma tale CDN può contattare altre CDN pagando qualcosa e agendo per conto del Provider, avendo una performance migliore.

Architettura di un Internet Data Center

Un Internet Data Center (IDC) è un luogo (un grande edificio ad esempio) che ospita fisicamente molte macchine server e si occupa della loro gestione, manutenzione e sicurezza. In particolare si occupa di hosting e housing. Per quanto riguarda la prima, fornisce, mantiene e amministra software e hardware per gestire i Web Server (nonché li dispone secondo una configurazione iniziale); per quanto riguarda la seconda, alloggia i server in un rack e li interconnette all'infrastruttura di connettività.
Esso dispone di una rete elettrica sofisticata con diverse linee di alimentazione indipendenti più gruppi elettrogeni di emergenza di vario tipo. Inoltre ha sistemi di condizionamento, sistemi antincendio, sistemi antiallagamento e sorveglianza 24 ore su 24.
Si connette all'esterno attraverso linee ad alta velocità ridondate, e possiede un AS (Autonomous System) e un ampio insieme di prefissi (peering multi-homed).

Dispone inoltre di firewall ridondati, di sistemi di intrusion detection, una lan per ogni customer e più lan per la gestione, switch e router accessibili solo via ssh (modalità sicura), un “Security Operation Center” (SOC) presidiato 24 ore su 24. Inoltre gode di controllo a ciascun livello della struttura, per ogni edificio e anche a livello di perimetro, con telecamere a circuito chiuso con videoregistrazione 24 ore su 24.
Ha anche un NOC, “Network Operatine Center”, per provisioning, monitoraggio, configurazione, identificazione e soluzione dei fault e rapporto con l’utenza, nonché di un Call Center con possibilità di scalare la chiamata al centro adeguato e per la gestione dei Trouble Ticket (TT), ovvero un sistema automatico di gestione delle domande (dubbi) degli utenti.

Interdomain Routing with BGP

La seguente è solamente una traduzione “paro paro” dalla dispensa impianti-bgp-protocollo-lab-11.pdf

INSTRADAMENTO INTERDOMINIO CON BGP

Sommario

- Introduzione all’instradamento
    - Instradamento intradominio
    - Instradamento interdominio
- Principi base di BGP
    - Semplici esempi
- Uno scenario complesso (un piccolo internet ma completamente sviluppato)
    - AS stub
    - AS stub multihomed
    - Piccola Internet

INTRODUZIONE ALL’INSTRADAMENTO

Perché l’instradamento?

- Ogni host (macchina) con una pila protocollare per la rete ha un problema di instradamento

        - Non devi essere per forza connesso ad una rete per aver bisogno dell'instradamento
- La pila protocollare di rete è in genere utilizzata per accedere ai servizi
    - Che possono essere locali (ad esempio, XWindows)

Lo strato ip

- Lo strato ip decide a quale interfaccia deve essere instradato un pacchetto in uscita
    - Gli host regolari hanno due interfacce, nic (network interface card) e loopback

Routers

- Un router:
    - Ha più di una scheda di interfaccia di rete (nic)
    - Ridà i pacchetti ip in entrata (che non sono per il router stesso) in pasto al processo di instradamento
        - Questa operazione è chiamata *trasmissione* o *inoltro*
    - Sono anche chiamati: *gateway*, *intermediate-system* (IS)

Protocolli di Instradamento

- I protocolli di instradamento sono usati per aggiornare automaticamente le tabelle di instradamento
- Essi ricadono in due categorie principali:
    - Protocolli di instradamento basati sullo stato dei collegamenti (link-state)
        - Approccio: parla dei tuoi vicini a tutti
        - Ogni router ricostruisce l'intero grafo della rete e calcola un albero dei cammini minimi per tutte le destinazioni
        - Esempi: is-is, ospf
    - Protocolli di instradamento basati sul vettore delle distanze
        - Approccio: parla di tutti ai tuoi vicini
        - Aggiorna le tue informazioni di instradamento basandoti su ciò che ascolti
        - Esempi: rip

Perché l'instradamento interdominio?

- Ogni organizzazione è una collezione di router e lan sotto una singola amministrazione

- Un algoritmo di instradamento (rip, is-is, ospf, …) può essere scelto per aggiornare automaticamente le tabelle di instradamento
- Quando alcune organizzazioni si uniscono per formare una internet (intesa più che altro nel senso più generico del termine) esse devono configurare i collegamenti fra di loro
    - Le LAN aggiunte sono chiamate “zone di demarcazione”

Che si dice sulle tabelle di instradamento?

- Per avere connettività globale:
    - Ogni router deve avere una voce (entry) di instradamento (possibilmente quella di default) che corrisponde all’indirizzo di destinazione del pacchetto
    - Questo dovrebbe essere vero sia per i pacchetti da consegnare localmente sia per i pacchetti da consegnare alle lan remote

Come aggiornare le tabelle di instradamento?

- In teoria hai tre opzioni
    - 1. Fai operare un singolo algoritmo di instradamento con le organizzazioni adiacenti
    - 2. Aggiorni le tabelle di instradamento a mano, aggiungendo rotte statiche per le lan esterne
    - 3. Combini un protocollo per gateway esterni (exterior gateway protocol) con il protocollo per gateway interni (interior gateway protocol) delle reti

1. Fai operare un singolo algoritmo di instradamento

- Era questo il modo quando venne introdotto egp (exterior gateway protocol). (Da non confondere questo specifico protocollo egp con la categoria generica di protocolli egp, di cui il protocollo egp fa parte. Purtroppo si chiamano allo stesso modo).

Singolo algoritmo di instradamento: problemi

- Tecnici
    - Lento a convergere
    - Tutte le organizzazioni sono costrette ad usare lo stesso algoritmo di instradamento
    - È difficile distribuire un nuovo algoritmo di instradamento
    - Difficile da “debuggare” e configurare
- Politici
    - L’instradamento è finalizzato a minimizzare l’uso globale delle risorse di rete

    - Non tiene conto di chi siano i proprietari dei collegamenti

2. Usa rotte statiche

- Approccio:
    - Nascondi la parte esterna della tua organizzazione
        - Nascondi le zone di demarcazione
        - Nascondi le altre organizzazioni

Rotte statiche

- Per ogni destinazione esterna:
    - Aggiungi una rotta statica a qualcuno dei router di confine
    - Stai dichiarando che puoi raggiungere tale destinazione mandando pacchetti al “next hop” specificato
    - Puoi usare le rotte di default
- L’algoritmo di instradamento igp propagherà all’interno della rete sia le destinazioni locali sia quelle aggiunte staticamente

Rotte statiche: problemi

- Tecnici
    - Difficile da aggiornare e da “debuggare”
    - I guasti (fault) non sono gestiti
        - Una rotta statica è disponibile anche quando il collegamento è interrotto (link down)
- Politici
    - Nessuna garanzia che il “next hop” sia disposto a consegnare il pacchetto

3. Protocollo per gateway esterni

- Approccio:
    - Nascondi la parte interna di tutte le organizzazioni

Rappresentare le destinazioni interne

- Ogni router esterno rappresenta le proprie destinazioni interne come se fossero locali
    - Puoi usare le rotte di default

Semplificare il grafo

- Semplifica il grafo
    - Considera la raggiungibilità esterna dei router

    - Considera la raggiungibilità interna dei router
    - Il grafo è in realtà gestito tramite connessioni tcp chiamate *peerings*

Risolvere il problema di instradamento

- Risolvi il problema di instradamento sul grafo semplificato
    - Basato su considerazioni politiche

Inserire le rotte

- Analogo al processo usato per le rotte statiche
    - Questa volta non è fatta alcuna assunzione sull'instradamento esterno!

Border Gateway Protocol

- Bgp è un protocollo di instradamento: mantiene aggiornate le tabelle di instradamento e propaga le informazioni di instradamento
- Tiene conto della disponibilità della disponibilità (o no) delle organizzazioni di cooperare nel processo (accordi commerciali, preferenze locali, priorità, problemi legali, …)

Un'esplorazione del mondo bgp

- Esploreremo il mondo di bgp passo passo
- Useremo esempi pratici per introdurre nuovi concetti
- Cominceremo dal bordo di Internet, supponendo di essere dei neofiti che si sono appena uniti al gioco
- Dopodiché considereremo la prospettiva degli ISP (Internet Service Provider)

PRINCIPI BASE DI BGP – L'Essenziale

Chi usa bgp?

- Bgp è usato da:
    - Clienti connessi ad un Internet Service Provider (ISP)
    - Clienti connessi a vari ISP
    - Clienti di transito
    - ISP che scambiano il traffico in un Punto di Interscambio (NAP – Neutral Access Point – Un punto di interscambio di dati diretto fra due o più ISP che si accordano per scambiarsi vicendevolmente il traffico delle proprie reti, piuttosto che pagare i costi di transito attraverso un tragitto più lungo rischiando anche il cosiddetto "single point of failure" rimanendo fuori da Internet qualora cadesse il link con l'upstream isp)
    - Clienti con reti molto grandi

Autonomous System Number (ASN – Numero di Autonomous System)

- Per usare bgp hai bisogno di un numero di identificazione chiamato *autonomous system number* (asn)
    - Tra 1 e 65.535 (due byte)
    - Numeri maggiori di 64.511 sono "privati"
        - Si vedrà più avanti perché ne abbiamo bisogno
- Puoi chiedere un asn a:
    - Asn globali – al tuo registro regionale di Internet (rir – *regional internet registry*): ripe, arin, apnic
    - Asn privati – al tuo upstream isp (gli upstream isp sono degli isp più grandi che concedono l'accesso a internet agli isp locali in maniera gerarchica)

Ci sono 2 bgp

- E-bgp
    - Bgp esterno
    - Single-hop (hop singolo)
- I-bgp
    - Bgp interno
    - Multi-hop (hop multiplo)
    - Importante: i-bgp non è un igp; piuttosto esso giace su un igp
- Ometteremo e- ed i- laddove sia evidente
- E-bgp
    - Usato da coppie di router in AS differenti per l'instradamento interdominio
- I-bgp
    - Usato da coppie di router all'interno dello stesso AS per permettere che le informazioni dell'instradamento interdominio attraversino l'AS
    - I router di uno stesso AS fanno peering i-bgp ciascuno con ogni altro (grafo completo)

Un peering bgp fra due AS

- Bgp permette ai router di scambiarsi informazioni solo se è attiva una sessione di peering
- Un peering bgp è la connessione tcp sulla quale sarà scambiata l'informazione sull'instradamento

Teoria e Laboratori

- La teoria è unita con dei laboratori

- Usiamo zebra
    - Demone di instradamento (i demoni sono dei processi figli del processo 1 “init”, ovvero dei processi che restano in memoria e sono fuori dal controllo dell’utente)
    - Funzionalmente molto simile a Cisco e ad altri demoni di instradamento proprietari

Vedi laboratori

Il Grafo di Internet (cosa sa Internet di sé stesso)

Molti recenti contributi alla ricerca sul “grafo di Internet”

- Gestione delle reti e pianificazione delle capacità
    - R. Govindan and A. Reddy – Un’analisi della topologia inter-dominio di Internet e della stabilità delle rotte – 1997
    - ……
- Identificazione delle relazioni commerciali sulla rete
    - Lixin Gao – Dedurre le relazioni tra Autonomous System in Internet – 2001
    - Z. Ge, D. Ratton Figueiredo, S. Jaiswal, L. Gao – La struttura gerarchica del grafo logico di Internet – 2001
    - L. Subramanian, S. Agarwal, J. Rexford, and R.H. Katz – Caratterizzare la gerarchia di Internet da vari punti di vista – 2002
    - G. Di Battista, M. Patrignani, and M. Pizzonia – Calcolare i tipi di relazione tra Autonomous System – 2002
- Dal punto di vista di un ricercatore di “scienze naturali”
    - A. Barabasi and R. Albert - l’emergere dello “scaling” nelle reti casuali – 1999
    - M. Faloutsos, P. Faloutsos, and C. Faloutsos – Relazioni “Power-Law” della topologia di Internet – 1999
- Generazione di reti casuali
    - A. Medina, A. Lakhina, I. Matta, and J. Byers - BRITE: Un approccio alla generazione della topologia universale – 2001
- ……
- *Algoritmi per esaminare i grafi*

… Che cos'è il "grafo di Internet"

- Obiettivo:
    - Mostrare le sorgenti dei dati per il grafo di Internet
    - Presentare alcuni dei sistemi di ricerca che si occupano del grafo di Internet
    - Fornire una chiave di interpretazione di alcuni retenci contributi alla ricerca sul grafo di Internet

Una storia a due livelli e un grafo a due livelli

Autonomous system (AS)

- Singola autorità amministrativa
- Ognuno identificato da un numero

Nascondere le parti interne delle organizzazioni:

- Una visione di Internet ad alto livello

Un grafo semplificato (quello formato dai router bgp)

Sorgenti dei dati di Internet – Cosa sa Internet di sé stesso

- Dati dei registri di instradamento di Internet (IRR – Internet Routing Registries)
    - RIPE (réseaux ip européens)
    - ARIN (american registry for internet numbers)
    - APNIC (asia pacific network information center)
    - …
    - Molti di essi replicati su RADB (Routing Assets DataBase)
    - Servizio "whois"
    - Linguaggio rpsl
        - Rfc 2622 e 2650; anche, rfc 2725 e 2726
- Dati operazionali (a livello di AS)
    - Tabelle bgp
        - Route Views – Università dell'Oregon
    - Annunci bgp
        - Routing Information Service – RIPE
    - Looking Glasses
    - Tabelle di Inoltro
- Dati operazionali (a livello di router)
    - Indagine topologica
        - Server traceroute
    - Performance
        - RIPE ncc "Test Traffic Measurements"

Quindi

- routing registries
- looking glasses
- route views
- routing info service
- topological probing

IRR – Internet Routing Registry

Unione di un numero sempre più grande di database sulle politiche di instradamento nel mondo che usa il linguaggio per la specifica di politiche di instradamento (Routing Policy Specification Language – RPSL)

Servizio Whois

È un servizio di tutti gli irr.

- Si può richiedere un whois su un indirizzo IP di una certa rete. Restituisce una breve descrizione testuale in forma libera dell'oggetto. Descrive informazioni per contattare gli amministratori.
- Si può richiedere un whois su un numero di AS (es. AS137). Ogni espressione sulle politiche di import è specificata usando un attributo "import". "Import: from AS5441 action pref=100 accept AS5441" significa che le rotte di AS5441 sono acetate da AS137 con preferenza 100. Lo stesso vale per l'attributo export per le politiche di export.

Sistemi intorno agli IRR – Hermes – Rappresentazione dei dati degli IRR

Hermes è un programma con varie funzionalità tra cui quella di rappresentare in forma grafica i dati degli IRR.

Looking Glass

Tipica domanda d'esame orale -Palla 09/01/2009 10:17

- Un Looking Glass è un software che gira sul Web Server di un ISP
- Permette ad utenti esterni di osservare il comportamento di instradamento e della rete all'interno della rete dell'ISP
- Di solito uno script perl
- Esegue comandi da shell, accede a router remoti, esegue ping, trace, o uno dei vari comandi di "show", permettendo la visualizzazione delle tabelle di instradamento IP e BGP

A volte compaiono delle inconsistenza tra Hermes e i dati forniti dagli IRR, poiché Hermes tende ad integrare tutte le informazioni ottenute dagli IRR.

Il progetto Route Views

È un progetto nato dall'Università dell'Oregon.
È largamente usato

- A scopi di manutenzione degli ISP
- Per la ricerca

Al momento

- Peering multi-hop ebgp in più di 40 punti intorno al mondo; più di 1,6 milioni di cammini
- In media più di 1800 connessioni al giorno

Route Views è un progetto che permette di vedere le informazioni mondiali BGP da punti di vista di altre locazioni. Inizialmente serviva agli ISP per sapere come le varie parti intorno al mondo vedevano le proprie reti, successivamente è divenuto molto importante anche in altri ambiti come la ricerca.
In pratica Route Views fa peering con molti punti intorno al mondo, e il comando "sh ip bgp <indirizzo-ip>" indica tutte le entry di tutti i router intorno al mondo che portano a quell'indirizzo IP. Ad esempio, per la rete 193.204.161.0, situata nell'AS 137, vengono restituiti tutti i percorsi che vanno da vari AS fino all'AS 137, creando quindi un albero la cui radice è 137 e le cui foglie sono i punti di partenza che portano all'AS 137. Ovviamente le informazioni includono anche il next hop, il neighbor che ha annunciato la rete, e i vari attributi, come per i comandi show ip bgp che si chiedono ai looking glass. La differenza è che con i looking glass si trova il singolo percorso (o la singola tabella) dal punto in cui si trova il looking glass (che è un router dove si è loggati in quel momento) fino all'indirizzo specificato. Con Route Views trovi tutti i percorsi da tutti i punti collegati a route views fino all'indirizzo specificato. Il grafo degli AS che viene fuori col looking glass è un semplice grafo con nodo sorgente (l'AS del looking glass) e nodo pozzo (l'AS della rete della query). Quello di route views è un albero, in cui le foglie sono le sorgenti e la radice è l'AS della rete della query. (In realtà del fatto che sia sempre un albero bisogna ancora discutere).

L'albero dunque indica il modo in cui il traffico in entrata raggiunge l'AS della rete specificata (nell'esempio l'AS 137).
In realtà ci sono solo pochi AS nell'albero. Ad esempio, mettiamo che manca l'AS 702. Significa che dall'AS 702 non possiamo raggiungere l'AS 137? No: semplicemente che l'AS 702 non è su alcun percorso che fa da un peerer di route views a 137. Potremmo ad esempio estendere l'albero con le informazioni ottenute dai looking glass, aggiungendo alcuni percorsi mancanti.

È sempre un albero?
Cosa posso fare se sono interessato a un grafo che illustri il traffico in uscita da 137 al resto del mondo?

- Una possibilità è

        - 1) Considerare tutti gli alberi (ognuno con un diverso AS come radice)
        - 2) Selezionare in ciascuno di essi il percorso da 137 alla radice (se ce n'è uno)
        - 3) Unire tutti questi percorsi
    - Un tale albero è grande e rappresentativo?

Ragionandoci sopra in realtà potrebbe anche non essere un albero. Mettiamo che per arrivare all'AS 10 c'è un percorso 1-3-10, un percorso 2-4-3-10 e un percorso 2-4-6-10: c'è il ciclo 3-4-6-10-3. Però route views mostra solo i percorsi migliori tra le route table, quindi nell'esempio precedente, dall'AS 4 il percorso scelto restituito sarà o quello per 3 o quello per 6, quindi l'altro ramo non esiste. Quindi? In realtà il grafo può benissimo non essere un albero a causa del forte livello di astrazione usato: stiamo considerando che ogni AS abbia un solo router esterno, quando in realtà un singolo AS può avere molti router esterni che indicano differenti rotte migliori.

Alcune informazioni danno righe del tipo "Dampinfo: penalty 3010, flapped 6 times in 00:30:13". Sono relative all'algoritmo di damping. Il damping di solito è un metodo che si usa per stabilizzare.
L'algoritmo di damping

- Punta a ridurre il raggio di propagazione dei "route flap" (cambi di rotta all'interno dei router)
- Requisiti
    - Convergenza veloce per i normali cambiamenti di rotta
    - Previsione dei comportamenti futuri
    - Soppressione delle rotte che oscillano
    - Promuovere le rotte stabili
- Aggiunge una penalità (penalty) per ogni flap
- Se la penalità supera un limite di soppressione: non pubblicizza la rotta ai peers

Il damping in pratica serve a contenere o eliminare le oscillazioni. La tecnica di damping in fisica prevede di applicare una forza proporzionale alla velocità ma in direzione opposta. In questo caso più c'è flap, ovvero più cambiamenti di rotta ci sono, più tali rotte guadagnano penalità, finchè la penalità non diventa abbastanza elevata da fermare i cambiamenti di rotta.

Routing Information Service

- Fornisce informazioni sull'instradamento bgp in maniera molto simile ai looking glass
- Importante: fornisce **informazione storica** sull'instradamento internet
- Colleziona informazioni di instradamento usando collezionatori remoti di rotte in diverse locazioni intorno al mondo e integra queste informazioni in una veduta esaustiva

In pratica il RIS preleva le informazioni dei vari router bgp e le mette nel suo database. Il database quindi contiene tutte le informazioni storiche delle tabelle dei vari router che fungono da sorgenti dati del DB.

- Esempio: il RIS permette di vedere come si poteva raggiungere un certo Web Server attraverso LINX il primo Gennaio 2002

- Esempio: il RIS permette di vedere in che modo uno specifico AS era raggiungibile dal punto di vista del collezionatore RIPE NCC nell'arco degli ultimi mesi

Ci sono due tipi di aggiornamenti BGP

- **Annunci:** "puoi raggiungere questo prefisso attraverso di me attraversando questo AS path"
- **Ritiri:** "non puoi più raggiungere questo prefisso attraverso di me"

Dato che ogni annuncio e ritiro avrà una data, il RIS potrà prelevare quest'informazione e usarla per stilare degli storici di raggiungibilità delle reti.
Esistono un numero fisso di Router del RIS che raccolgono informazioni (di nome rrc00, rrc01, rrc02 e così via. Ad esempio rrc00 è il RIPE NCC sito ad Amsterdam mentre rrc01 è il LINX di Londra). La richiesta che si può fare è chiedere ad esempio come uno di questi router o un suo peer (perché i router del RIS fanno peer con vari router intorno al mondo, proprio come nel caso di Route Views) poteva raggiungere un certo AS da una certa data ad un'altra.

BGPlay non è argomento d'esame -Palla 12/01/2009 14:25

Un sistema con interfaccia grafica che si basa sul RIS è BGPlay. Esso ti dice in maniera grafica e anche con animazioni come cambiano i percorsi dai peerer del RIS fino ad un AS specificato. In particolare ogni frame dell'animazione corrisponde ad un annuncio, un ritiro o un cambio di AS path.

Per rappresentare i cammini bgplay usa diversi colori se essi hanno qualche ramo in comune, così da poter distinguere i due cammini differenti.
Semplificare il grafo:
Spesso, i cammini possono essere "fusi" senza causare ambiguità
Infatti:

- Se partizioniamo i cammini in due gruppi tali che ogni gruppo formi un albero, allora i cammini di un gruppo possono essere uniti
- Ad ogni gruppo è assegnato un colore distinto

Questa buona spiegazione l'ho trovata sul forum:
su k-tree partition il prof non ha detto molto... bgplay proietta l'insieme degli AS-path associati ad un prefisso; per una query su ad esempio 193.204.0.0/15 con un intervallo temporale utile può verificarsi che gli AS-path siano moltissimi; da questo segue il problema di visualizzare tutti gli AS-path senza però perdere potere espressivo. L'idea allora è quella di effettuare un merge di cammini che in nessun modo possano formare un ciclo ed assegnare a questo insieme di cammini (anche l'isieme formato da un solo cammino è valido) un colore distinto dagli altri. Tale azione estesa a tutti i cammini comporta il partizionamento dell'albero dei cammini originati dall'AS (nell'esempio il 137) che annuncia il prefisso (193.204.0.0/15). Il problema è riconducibile a k-colorability ma sfortunatamente esso è NP-hard.
La soluzione finale è stata quella di implementare una euristica in grado di partizionare l'albero dei cammini. Il maccanismo di funzionamento è il seguente:
- all'istante iniziale si crea un bucket (cestino) rappresentante l'insieme dei cammini di un certo colore; tale bucket conterrà un insieme di cammini il cui merge non produce cicli.

- si prende un cammino e lo si assegna ad un bucket (in ordine sequenziale); se il merge degli AS-path del bucket con il nuovo cammino produce un grafo aciclico allora l'assegnazione del colore a quell'AS-path è andata a buon fine altrimenti si prova con un altro bucket. Se alla fine nessun bucket e andato bene allora si crea un nuovo bucket che rappresenta un altro colore.

Livello dei Router

A questo livello ci sono le tecniche di "topological probing". Esistono sistemi basati su tecniche simili a quelle descritte in R. Govindan and H. Tangmunarunkit - Heuristics for Internet Map Discovery – 2000.
Esempi: *skitter* e *nethunter*
Skitter effettua dei traceroute da una specifica fonte
Nethunter e rocketfuel effettuano dei traceroute da certe fonti chiamate traceroute server

Conclusioni

Ci sono molti grafi di Internet

- Livello degli AS contro livello dei router
- Registri contro Dati Operazionali
- Punto di vista storico

Riepilogo

- ***Routing Registries***
  - Whois
  - **Hermes**
- ***Looking Glass***
- ***Route Views***
  - Albero Route Views
  - Algoritmo di Damping/Flapping
- ***Routing Information Service***
  - **BGPlay**
- ***Topological Probing***
  - **Skitter**
  - **Nethunter**

Individuare le relazioni Customer-Provider fra gli Autonomous System

Il problema è introdotto da Lixin Gao.
Le relazioni sono classificate in tre categorie:

- customer-provider
- peer-peer
- sibling-sibling

Come input sono utilizzate le tabelle di instradamento BGP
Le euristiche sono verificate con informazioni provenienti da altre fonti.

A prima vista le relazioni peer-peer sono tra due AS che si scambiano regolarmente il traffico a vicenda. In particolare ciascuno può inviare all'altro il traffico originato in sé stesso o da AS a livello più basso che sono propri customer.
Le relazioni customer-provider sono fra due AS su differenti livelli. Il customer può inviare al provider il traffico originato in sé stesso o da propri customer; il provider può inviare al customer qualsiasi traffico.
Il sibling-sibling potrebbe essere una relazione tra due AS stub che seppur collegati fra loro non hanno bisogno di scambiarsi traffico perché questo potrebbe passare da qualche provider in comune. In realtà questa per ora è solo una mia ipotesi.

Se tutti gli AS rispettano le regole di annuncio, gli AS path dovrebbero essere tutti *validi*.
Un AS path valido è di due tipi:

- un'eventuale salita (formata solo da rami che vanno dal customer al provider di una relazione customer-provider) seguita da un'eventuale discesa (formata solo da rami che vanno dal provider al customer di una relazione customer-provider)
- un'eventuale salita seguita da un arco peer-peer seguito da un'eventuale discesa

Il problema Tipe of Relationship (ToR)

Problema ToR

Dato un grafo non orientato G e un insieme di cammini P, dai un'orientazione ad alcuni degli archi di G tale da minimizzare il numero di cammini invalidi in P.

I cammini *non validi* sono quelli che non sono "valley free" (cioè quelli che compiono più di una salita e/o discesa), e quelli che comprendono più di un peering (insomma tutti quelli che non sono validi, eggià!)

Si ipotizza che il problema ToR sia NP hard.

Nota: il problema ToR non prevede di sapere a priori le relazioni fra gli AS. Anzi, esso serve a trovarle. Ad esempio, costruito un grafo degli AS e vari cammini che vengono effettivamente seguiti, il tutto grazie alle informazioni di Route Views, se risolviamo il problema ToR troviamo un'orientazione, e grazie a questa speculiamo sulle relazioni fra gli AS.

Un nuovo contributo

Mostriamo che, sebbene il problema ToR sia NP-hard, possiamo trovare una soluzione senza cammini invalidi (se esiste) in tempo lineare.
Proponiamo delle euristiche per il problema generale basata su un nuovo paradigma e mostriamo la loro efficacia contro gli insiemi di dati disponibili pubblicamente.
Gli esperimenti mettono in evidenza che le nostre euristiche rendono significativamente meglio delle euristiche dello stato dell'arte.
Indipendentemente dal nostro lavoro, Erlebach, Hall, e Schank del Gruppo della Teoria delle Reti di Telecomunicazioni hanno scoperto risultati analoghi riguardo la complessità temporale del problema generale e la linearità nel caso di tutti cammini validi.
Tuttavia, mentre loro mettono più enfasi sull'approssimabilità del problema, noi ci focalizziamo più sulla progettazione e sperimentazione di un effettivo approccio euristico.

Il problema di ottimizzazione ToR corrisponde al seguente problema di decisione.
Problema ToR-D

Dato un grafo non orientato G, un insieme di cammini P, e un intero k, vedi se è possibile dare un'orientazione ad **alcuni** degli archi di G tale che il numero di cammini non validi in P sia al più k

Nelle dispense viene seguita la strada che cerca di risolvere il ToR-D con k=0 (almeno così ho capito).

SAT è un argomento di Informatica Teorica (slide di riferimento impianti-bgp-relazione-fra-as pag.10)-Palla 12/01/2009 15:28

Non chiede all'esame la parte dell'euristica e tutto ciò che riguarda Informatica teorica (NP-hard) -Palla 13/01/2009 10:20

Osservazione

Un cammino p = v1, …, vn è valido se e solo se non ha un vertice vi tale che i due archi di p incidenti in vi sono diretti fuori da vi.

Basandosi su questa osservazione il problema può essere mappato a 2SAT

Dato un insieme X di variabili booleane e una formula in forma normale congiuntiva composta da clausole di due letterali, dove un letterale è una variabile o una variabile negata, trova un'assegnazione di verità per le variabili booleane in X tale che la formula sia soddisfatta.

Esempio (x1 || x2) && (!x2 || !x3) && (x3 || !x1)
Mappare il problema a 2SAT

- associa ogni arco ad una variabile booleana

- assegna ad ogni arco un'orientazione arbitraria (diciamo casuale)
- un'assegnazione di verità per le variabili booleane corrisponde ad un'orientazione per tutti gli archi del grafo degli AS (e viceversa)
    - se la variabile è VERO, l'arco associato preserva la propria direzione originale
    - se la variabile è FALSO, l'arco associato è invertito
- costruire la formula 2SAT è semplice: per ogni nodo vi con due archi orientati incidenti, metti una nuova clausola nella formula; come letterali ci vanno le variabili corrispondenti ai due archi, normali se entranti nel nodo, negate se uscenti dal nodo

La soluzione del 2SAT ci darà la soluzione del problema.

Risolvere un grande 2SAT

Data una formula 2SAT calcola il grafo dei letterali
(ci sono tutti i letterali possibili. Una disgiunzione equivale a due implicazioni. Ogni implicazione equivale a un arco orientato nel grafo dei letterali. Se c'è un cammino che porta da una variabile alla sua negazione e viceversa, non c'è soluzione per il problema 2SAT)
C'è un cammino da un letterale al suo opposto e viceversa (nessuna soluzione per il problema 2SAT) se e solo se non esiste una soluzione con tutti cammini validi (nessuna soluzione per il problema ToR-D con k=0)
Componenti Fortemente Connesse
Calcola le componenti fortemente connesse del grafo dei letterali.
Componente fortemente connessa = insieme massimale di vertici tale che per ogni coppia di vertici (u,v) dell'insieme, esiste un cammino diretto da u a v e viceversa (cioè ogni nodo può raggiungere ogni altro nodo nell'insieme).
Una soluzione al problema 2SAT non esiste se e solo se i due letterali della stessa variabile booleana cadono nella stessa componente connessa (ovvio).
Il test prende un tempo dell'ordine O(n+m+q) dove n è il numero degli AS (clausole in 2SAT), m è il numero degli archi (variabili booleane in 2SAT) e q è la somma delle lunghezze degli AS-path (in 2SAT non credo abbia riscontri, in effetti ancora non si capisce bene come funziona la conversione del problema, cioè come l'insieme P dei cammini di ToR influisca sulla trasformazione in 2SAT).

L'esperimento effettuato

Sono stati usati dieci looking glass telnet come fonti di dati.
Sono stati uniti tutti gli AS-path per testare l'algoritmo.
Sono state usate quattro fonti del web per convalidare l'output.

La fonte più completa è l'Università dell'Oregon, che è anche l'unica con la quale non si riesce a ottenere l'orientabilità con tutti cammini validi. In tutti gli altri casi si è riusciti ad orientare il grafo senza creare cammini non validi, a convalida del fatto che le relazioni fra gli AS sono pressoché fissate e ricavabili.

Calcolo di una soluzione senza cammini invalidi

- considera il grafo orientato aciclico delle componenti connesse del grafo dei letterali
- calcola un ordinamento topologico delle componenti connesse e assegna un intero ad ogni componente

- chiama f(x) l'indice della componente alla quale appartiene il letterale x
- se f(x)>f(!x) assegna VERO a x, altrimenti assegnagli FALSO

Nota

Esiste il problema MAX2SAT che non è un problema di decisione bensì di ottimizzazione: infatti dice "trova un'assegnazione di verità alle variabili booleane tale da soddisfare il maggior numero di clausole". Se il problema ToR-D con k=0 è mappabile a 2SAT, a MAX2SAT è mappabile proprio lo stesso problema ToR. Tali problemi sono comunque NP-hard (nota: non è ancora chiarissimo come il ToR si mappi su MAX2SAT perché non viene fatta menzione di come si usino i cammini nella trasformazione).

Il nostro approccio euristico

Grandezza del problema
3,4M AS-path
10K vertici
24K archi
(ne viene che 3,1M di AS-path sono stati sistemati nel primo passo dell'algoritmo seguente, e si arriva quasi ai totali 3,4M col secondo passo)

L'approccio è molto semplice:

- Trova un insieme molto grande di AS-path che ammettono un'orientazione
- Prova a reinserire gli AS-path esclusi

1) Trova un insieme molto grande di AS-path che ammettono un'orientazione
   1.a) Dai un rank agli archi a seconda di quanti cammini passano per essi
   1.b) Costruisci il grafo dei letterali
   1.c) Calcolane le componenti fortemente connesse
   1.d) Trova tutte le variabili per le quali entrambi i letterali cadono nella stessa componente fortemente connessa
   1.e) Considera gli archi corrispondenti a quelle variabili e elimina quello con rank più alto, e anche tutti i cammini che lo usano
   1.f) goto 1.a)
2) Prova a reinserire gli AS-path rimasti fuori
   2.a) x=1
   2.b) aggiungi x cammini
   2.c) if (è ancora risolvibile)
   Commissiona l'aggiunta dei cammini
   x=x*2
   Else
   If x=1
   Scarta il cammino (si deduce che stavolta sarà scartato per sempre)
   x=x/2
   2.d) goto 2.b)

Si verifica che l'approccio SAT lavora meglio dell'approccio SARK che è lo stato dell'arte. Dalle varie tabelle sulla dispensa si intuiscono varie altre cose. È bene darci un'occhiata.

E per le relazioni peer-peer? Sembra che con questo approccio vengano totalmente escluse. Sulle dispense si parla di Grafo delle Differenze. La pagina recita: abbiamo inserito un arco

non orientato (e due eventuali vertici) nel grafo per ogni arco orientato in maniera opposta. Mi chiedo: in maniera opposta a cosa? E dove è stato inserito questo arco? E questi eventuali vertici? Per arco orientato in maniera opposta si intende per caso il cambiamento dell'orientazione quando risolviamo 2SAT? Ai posteri l'ardua sentenza.
Poi si parla esplicitamente delle relazioni peer-peer. Si intuisce che esse vengano cercate solo dopo aver costruito il grafo con il precedente approccio. Dice che scoprire le relazioni peer-peer è un problema difficile al quale ci si riconduce addirittura da MAX-INDEPENDENT-SET. Nelle tabelle seguenti si nota che il numero delle relazioni peer-peer cambia notevolmente a seconda della fonte di dati. Probabilmente c'è bisogno di maggiori informazioni di tipo semantico per poter procedere con un'accurata selezione delle relazioni peer-peer.

Conclusioni
Abbiamo mostrato che ToR è NP-hard ma…
Se esiste una soluzione senza cammini invalidi, può essere trovata in tempo lineare mappando il problema a 2SAT.
Proponiamo delle euristiche per il problema generale basato sulla riduzione a 2SAT.
Mostriamo l'efficacia delle euristiche contro gli insiemi di dati disponibili pubblicamente.
Gli esperimenti mettono in evidenza che le nostre euristiche operano molto meglio delle euristiche dello stato dell'arte.
Mostriamo che scoprire le relazioni peer-peer è un problema difficile se si vuole massimizzarle.

Problemi
Come determinare le relazioni peer-peer una volta che il grafo è orientato?
Ogni AS-path è soppesato senza tener conto della grandezza del suo prefisso. Fino a che punto è corretto?
Degli snapshot delle stesse tabelle BGP prese in diverse date aiuterebbero a capire meglio le relazioni fra gli AS?
Cosa succederebbe se conoscessimo in anticipo l'orientazione di alcuni archi?
(Insomma, a quanto pare si può anche non saperlo come trovare le relazioni peer-peer…).

Analisi dei Dati di Internet

Perché analizziamo i dati su Internet?
Motivazioni pratiche:
- La tecnologia reggerà con la crescita di Internet?
- Possiamo gestirlo meglio?
- Internet quanto è tollerante ai guasti?
- Dov'è il miglior posto in cui dispiegare i nostri web server?

Motivazioni scientifiche:
- Possiamo caratterizzare Internet (topologia e crescita)?
- Perché Internet è così?
- Potrebbe essere migliore?

Vengono illustrati alcuni grafici, dati e analisi (dispensa impianti-bgp-data-analysis-03.pdf) presi da varie fonti.

Crescita delle Tabelle di Instradamento BGP

Il problema
I router di dorsale hanno grandi tabelle di instradamento.
La ricerca nella tabella di instradamento è effettuata per ogni pacchetto inoltrato.
Il tempo di ricerca cresce con il crescere della tabella di instradamento.
Le tabelle di instradamento diventeranno troppo grandi?

Rotta: prefisso più informazioni di instradamento (esempio interfaccia di uscita, next hop, as path)
Le Tabelle di Instradamento sono liste di rotte.

- Forwarding Information Base (FIB): usato dallo strato ip per inoltrare realmente i pacchetti
- Routing Information Base (RIB) : usato dai demoni di instradamento per operare il protocollo

RIB vs FIB

- RIB contiene più informazioni per ogni rotta (ad esempio, l'AS path in bgp)
- Solo una selezione di RIB diventa FIB

Zona "Default Free"
Insieme di router che hanno una tabella di instradamento che non contiene una rotta di default.
Sono i router più stressati di Internet (i più grandi FIB e RIB).
Fondamentalmente, sono le dorsali.

Tramite un grafico che illustra la crescita del numero di entry bgp nelle tabelle di instradamento dei router bgp nel corso degli anni, ci si rende conto che l'introduzione della tecnica CIDR (che ricordiamo, annulla le vecchie "classi" degli indirizzi ip, consentendo di poter rappresentare con uno stesso indirizzo ip e una maschera, sia la rete che l'host, decidendo quanti bit riservare alla rete e quanti all'identificazione dell'host) è stata fermata la crescita esponenziale che stava assumendo il numero di entry nelle tabelle di instradamento. Dopodiché, a metà del '98 è ricominciata la crescita esponenziale (dopo il CIDR).

Famosa è anche la legge di Moore (cofondatore della Intel), che ha osservato che la densità dei transistor sui chip raddoppia ogni anno. Dopo qualche anno è stata osservata una piccola deviazione verso il basso (in fondo era pur sempre una legge empirica). Al momento la potenza delle cpu raddoppia ogni 18 mesi. Si osserva che le entry BGP (FIB) crescono in maniera meno rapida.

Margini di sicurezza
Il tasso di crescita corrente è circa metà della legge di Moore, tuttavia…
Sembra che il tasso di crescita stia accelerando avvicinandosi ad una crescita iperesponenziale!

In questo caso la crescita delle tabelle di instradamento alla fine sopravanzerà sulla legge di Moore.
…o forse la legge di Moore potrebbe fallire, a causa dei limiti fisici.
Che il tasso di crescita tendi ad un iperesponenziale lo mostrano i grafici ottenuti dall'analisi dei dati.

Una previsione pessimistica ci diceva che i ritmi della crescita sarebbero stati confermati fino ad arrivare a 150.000 entry all'inizio del 2002, una previsione ottimistica ci diceva che invece saremmo ridiscesi a 75.000 grazie all'eliminazione delle "rotte superflue". Com'è andata? Grazie alle tecniche CIDR e di aggregazione è stata arginata un po'. Ora i ritmi di crescita pare stiano ricrescendo in maniera molto rapida. Gli ultimi dati ci dicono 200.000 entry in ottobre 2006.

Rotte Superflue
Una rotta superflua può essere rimossa da una tabella di instradamento senza compromettere la connettività.
Rimuovere tale rotta significa riconfigurare i dispositivi in modo tale che le cose funzionino senza quella rotta.
Rimuovere una rotta superflua può richiedere uno sforzo significativo.

Aggregazione
Due rotte che possono essere aggregate (CIDR)
193.204.16.0/24 e 193.204.17.0/24 possono essere annunciate come 193.204.16.0/23 (che è una rotta invece di due)
E sugli AS-path?
Gli AS di origine possono sempre effettuare un'aggregazione
Gli AS di transito hanno bisogno di controllare la consistenza degli AS-path e usare correttamente gli attributi bgp (AS-SET e AS-SEQUENCE), ma questo è inusuale.

Prefissi radice
In un insieme di prefissi possiamo identificare una gerarchia di prefissi (alcune potrebbero essere banali). 6.10.0.0/16 è il prefisso radice di 6.10.1.0/24 (e di tutti i prefissi di cui quest'ultimo è a sua volta radice). In questo modo si forma un albero, con varie profondità gerarchiche, e una radice, per ogni gerarchia. Un instradamento perfettamente gerarchico e l'aggregazione CIDR condurrebbe le tabelle di instradamento a sole entry con prefissi radice (il nostro obiettivo ideale).
Però una rotta più specifica potrebbe essere superflua ma potrebbe anche non esserlo.

- errori di configurazione (superfluo!)
- multihoming (ridondanza + bilanciamento del carico) (necessario)
- sommarizzazione
  - violazione della gerarchia dell'instradamento: per liberarcene dovremmo rinumerare parte della rete

(l'ultimo punto non l'ho ben capito. La sommarizzazione è un sinonimo di aggregazione, è proprio la stessa tecnica. E l'aggregazione annulla gli indirizzi più specifici, di certo non li crea. Comunque la sommarizzazione/aggregazione di certo viola la gerarchia di instradamento, nel senso che noi potremmo volere che i pacchetti destinati ad un indirizzo più specifico vengano inoltrati su un canale, e i restanti pacchetti per il prefisso radice vengano inoltrati su un altro. Non ho capito però perché dovremmo "liberarcene". Ipotesi: per

raggiungere quello stato ideale in cui le tabelle di routing siano solo con prefissi radice. Allora sì, dovremmo assegnare degli indirizzi diversi alle varie reti, sempre che si intenda questo per "rinumerare", ma io ne dubito fortemente. E poi a che scopo rinumerare? In questo modo non ci sarebbe comunque una significativa riduzione delle entry nelle tabelle! Bah, quanto sono poco chiare queste dispense…).

Le tecniche usate sono servite per arginare un po' la crescita delle entry, ma questa continua comunque a rimanere preoccupante.
Nota: finchè tale tasso di crescita rimane sotto la legge di Moore, significa che anche se il numero di entry aumenterà, ci sarà sempre abbastanza potenza di calcolo atta a sostenere tale aumento. Se si arriverà al punto che la crescita delle tabelle supererà il tasso di crescita della potenza dell'hardware, si arriverebbe prima o poi al punto che non ci saranno più cpu tali da poter garantire in tempo utile efficienti ricerche su tabelle di quelle dimensioni.

Crescita dell'allocazione degli indirizzi

Il problema
La crescita globale delle tabelle di instradamento bgp può essere causata da…

- richieste di spazi di indirizzi (esempio, il mercato)
- questioni tecnologiche (esempio, multihoming)
- politiche di allocazione degli indirizzi

Attualmente RIPE e ARIN detengono la maggior parte degli indirizzi IP, APNIC di meno.

Spazio degli indirizzi annunciato
È il numero di indirizzi coperto da tutti i prefissi. Non ci sono abbastanza dati da capire se esso stia crescendo in maniera lineare o esponenziale.
C'è anche il numero di AS annunciati

Alcune considerazioni
Sembra che Internet continui a crescere.
Il lento abbassamento della crescita delle tabelle di instradamento è dovuto alla migliore gestione delle rotte annunciate.
Si può resistere per sempre?
C'è bisogno di maggior comprensione della crescita.

<u>L'opinione del saggio</u>
Circa il 70% delle tabelle di instradamento globali default free è dovuto al multihoming. –G. Huston
Le rotte per gli AS multihomed sono il 30%. – K. Claffy

Sfortunatamente non ci sono dati disponibili al pubblico sui fenomeni del multihoming e la loro crescita, anche se i dati si possono ricavare da altre fonti, ad esempio i RIB bgp del progetto Route View dell'Oregon.

Il problema no è la grandezza delle tabelle, ma il numero di update per secondo. – G. Huston

Ma i problemi sono strettamente correlati

- meno rotte portano a meno update
- l'aggregazione nasconde parte delle oscillazioni delle rotte

Così il rimedio è sempre lo stesso: migliore amministrazione degli indirizzi

Internet dal Punto di Vista delle Scienze Naturali

Caratteristiche invarianti
Internet è in continua trasformazione
Come possiamo caratterizzare il grafo degli AS di Internet?
C'è qualcosa in Internet che NON cambia (un'invariante) ?

Grado degli AS
Il grado di un AS è il numero di AS adiacenti ad esso.
Per ogni valore di grado, calcola il numero (#) degli AS che hanno questo grado.
Traccia un punto per ogni coppia (grado, #)
Usa una scala log-log (scala in logaritmo di x e y)
Otteniamo una "power law" ($y=1/x^a$ dove y è # e x è grado), come di vede dall'andamento lineare (infatti le power law appaiono sempre lineari in scala log-log).
La power-law ha una cosiddetta distribuzione "scale-free", ovvero la media dei valori non descrive il fenomeno.
Leggi analoghe già viste sono la legge di Zipf e le distribuzioni heavy-tailed.

La (in)stabilità di BGP

Instabilità di bgp
Data una configurazione bgp dei router di una rete, l'instradamento è stabile?
Bgp può causare oscillazioni nell'instradamento?
Ci sono configurazioni "sicure"?
C'è un modello semplificato che possiamo usare per studiare la stabilità dell'instradamento prodotto da bgp?

Usando un modello semplificato come sulle dispense impiangi-bgp-modello-09.pdf in cui si disegna ogni volta prima gli update dei router e poi i cambiamenti dell'instradamento, si nota che ci sono continue oscillazioni nell'instradamento. In quell'esempio, ogni router ha nella propria tabella una serie di rotte consentite tra cui scegliere (se arrivano gli annunci giusti). Se arrivano più annunci tra quelli disponibili, il router sceglie come rotta quello che sta più in alto. Ad ogni step arrivano annunci diversi perché ad ogni step i router cambiano le rotte, e ad ogni step i router cambiano le rotte perché ad ogni step arrivano annunci diversi. Per semplificare il modello ovviamente tutto ciò è stato discretizzato, si assume che i tempi dei router siano sincronizzati, che tutte le modifiche avvengano nello stesso momento, così come la partenza e l'arrivo di tutti i pacchetti bgp.

Fondamentalmente, la tempistica ha un ruolo fondamentale.

Si osserva che nell'esempio, basta eliminare una delle rotte disponibili per uno dei router, per assicurare che ci si possa portare ad uno stato stabile. In questo caso si osserva che l'instradamento può raggiungere uno stato stabile indipendentemente dal tempismo degli update.

Quindi per "configurazione bgp" si intende un insieme di rotte disponibili per ogni router tra cui scegliere in base agli annunci ricevuti (nell'esempio l'AS 0 non dispone di questo insieme, perché è quello "da raggiungere". In pratica una configurazione la stiamo intendendo per ora relativa esclusivamente ad una destinazione che origina alcuni blocchi di indirizzi). Una configurazione bgp può avere anche vari stati stabili possibili.

Si dice **struttura disagree** una configurazione di 3 AS a triangolo, di cui uno, chiamiamolo AS 0, è quello destinazione, e gli altri due hanno come disponibili entrambe le rotte che portano all'AS 0, sia quella diretta, sia quella che passa dall'altro.
Nell'esempio si nota che la struttura disagree può portare a ben due stati stabili differenti, ma che se avesse update perfettamente sincronizzati come nell'esempio precedente, non porterebbe ad alcuno stato stabile.

Stiamo parlando di configurazioni realmente esistenti: la configurazione disagree è possibile in vari scenari. Diamo una descrizione RPSL a tre scenari possibili.

1. Ciascuno dei due AS ha una politica per la quale, se importa annunci da 0 e da 2, quelli che vengono da 2 sono preferiti;
2. Uso del costrutto rpsl "community" che non ho assolutamente capito a cosa serve;
3. Annuncio da parte dell'AS 0 con prepend (cosicché venga preferito l'altro percorso).

Stati stabili multipli possono portare a degli "inneschi di rotta". Ad esempio, se si ha uno stato stabile con un link di backup che non si usa, e poi andasse giù il link principale, si comporrebbe un'altra configurazione. Dopodiché viene ristabilito il link principale. Il successivo stato stabile che si raggiunge può non essere uguale a quello iniziale.

Un Modello per BGP – Definizione Del Modello

Si ha bisogno di un modello per bgp per studiare in maniera statica un mondo ben più dinamico, per considerare "tutte" le possibili tempistiche, e con un'assunzione che semplifichi di molto il nostro modello: trattiamo solo con gli AS-path. Comunque generalizzare non è difficile.

Un'istanza BGP

- Grafo di nodi e archi
    - I nodi rappresentano i router di confine che parlano BGP, gli archi rappresentano i peering; i nodi sono numerati da 0
- Nodo **0**, chiamato **origine**
    - Rappresenta l'AS che origina qualche blocco di indirizzi

- Per ogni nodo i, un insieme **permitted(i)** di **cammini permessi** per l'origine
  - Contiene almeno il cammino vuoto (Φ): significa che 0 è irraggiungibile
- Un **ranking**, ovvero una classifica di cammini permessi per ogni nodo
  - I cammini sono ordinati secondo le politiche
  - Il cammino vuoto è sempre l'ultimo in classifica

Stato di Instradamento del Router

- Uno **stato di instradamento** s_i di un router i è una (n+1)-upla $<p_{i0}, p_{i1}, p_{i2}, \ldots, p_{in}>$, dove
  - n è il numero di router non-zero
  - p_{ij} appartiene a permitted(i) e j denota un router
  - se i e j non sono peerer (non sono adiacenti), allora $p_{ij} = \Phi$
  - se $i \neq 0$, allora $p_{ii} = \Phi$
  - $p_{00} = 0$ e $p_{0j} = \Phi$ per $j \neq 0$
- Lo **stato iniziale di instradamento** è
  - Di un router non-zero è $< \Phi, \ldots, \Phi >$
  - Di 0 è $<0, \Phi, \ldots, \Phi >$
- Il cammino p_{ij} rappresenta come raggiungere, attraverso il vicino j, il blocco di indirizzi annunciato da 0
- Il cammino corrente **best(i)** usato per raggiungere 0 è il più alto in classifica tra quelli di p_{ij}

Stato di instradamento del Sistema

- Lo **stato di instradamento** S di un sistema è una (n+1)-upla $S=<s_0, s_1, \ldots, s_n>$, dove
  - n è il numero di router non-zero
  - s_i è lo stato di instradamento del router i
- Nello **stato iniziale di instradamento** del sistema tutti i router sono nello stato iniziale di instradamento
- Poiché S è una (n+1)-upla di (n+1)-uple, può essere rappresentato da una matrice (n+1)x(n+1) di cammini, dove ogni riga rappresenta lo stato di un router
- Chiamiamo tale matrice **matrice di stato**

Si osservi che best(i) (con i=1, …, n) rappresenta la sola porzione dello stato del sistema che influenza l'inoltro dei pacchetti
Negli esempi sulle dispense, la matrice ha caselle col carattere rosso quando i valori non possono cambiare (saranno sempre quelli), e caselle col carattere nero che sono i valori che possono cambiare durante gli step di un'analisi.

Transizioni di Stato

- Il Sistema può cambiare il proprio stato a partire da quello corrente
- Una **transizione** (**valida**) $T:S' \rightarrow S''$ dallo stato S' allo stato S'' è tale che per ogni i, j (i=1, …, n; j=0, …, n) accade una delle seguenti cose:
  - O $p''_{ij} = p'_{ij}$ (i non riceve update da j)
  - Oppure
    - Sia k tale che p'_{jk} = best(j) (il miglior cammino nella riga j) e supponiamo che i sia un peerer di j
    - Se $p'_{jk} = \Phi$ (i riceve una ritiro da j), allora $p''_{ij} = \Phi$
    - Se $p'_{jk} \neq \Phi$ (i riceve un annuncio da j), allora

- Se ip'_{jk} è in permitted(i), allora $p''_{ij} = ip'_{jk}$
- Altrimenti $p''_{ij} = \Phi$

Sembra complicato ma dopo aver analizzato bene le formule diventa banale.
Da S' a S'' laddove non ci sono update, le p dei router rimangono uguali. I router che ricevono update, se si tratta di annunci aggiornano il fatto che possono passare dal router che ha fatto l'annuncio se quel percorso lo hanno fra quelli permessi, se si tratta di ritiri aggiornano il fatto che non possono passarci.

Transizioni
Si osservi che la definizione di transizione valida consente di considerare un'ampia varietà di possibili tempistiche del sistema. In che senso? Nel senso che non dobbiamo considerare come un annuncio o un ritiro ogni cambiamento di rotta migliore per un nodo. Prendiamo come esempio la struttura disagree: abbiamo visto che applicando un perfetto sincronismo non giungeremo mai alla stabilità, invece applicando le transizioni opportune possiamo scegliere di evitare di fare entrambi gli annunci, in modo da portare il sistema alla stabilità. Inoltre, si noti che le transizioni potrebbero portare un router in uno stato tale che la sua scelta migliore è peggiore di quella del suo precedente stato (cioè **non** possiamo dire che le scelte migliori prendono via via cammini sempre più alti in classifica in maniera non decrescente).

Un Modello per BGP – Stabilità

Stato di instradamento stabile
Proprietà (*update ricevuti*): se una transizione T:S'→S'' è tale che $best_{S''}(i) \neq best_{S'}(i)$ per qualche i=1,…n, allora esiste un nodo j tale che $p''_{ij} \neq p'_{ij}$ e j è un neighbor di i (cioè, il nodo i deve aver ricevuto un update).

Stato di instradamento strettamente stabile
- Data un'istanza BGP, uno stato di instradamento è **strettamente stabile** se, per ogni transizione valida T:S→S', si ha che S'=S
- In tale stato
  - … la matrice di stato non cambia
  - … le transizioni possono ancore verificarsi, ma solo a causa di:
    - Assenza di aggiornamenti (transizione banale)
    - Aggiornamenti ripetuti

Stato di instradamento a inoltro-stabile
- Data un'istanza BGP, uno stato di instradamento è a **inoltro stabile** (forwarding-stable) se, per ogni transizione valida T:S→S', si ha sempre che:
  - $best_{S'}(i) = best_{S}(i)$ per ogni i = 0, …, n (ovvero annunci e ritiri potrebbero anche modificare alcuni valori della matrice, ma le rotte migliori rimangono le stesse)
- Proprietà: La stabilità stretta implica la stabilità d'inoltro, e non è vero il contrario

Lemma

- Sia S uno stato in stabilità di inoltro; non esiste alcuna coppia di transizioni consecutive T':S→S', T'':S'→S'' tale che
    - $best_{S''}(i) \neq best_S(i)$ per qualche i = 0,…,n
- Dimostrazione (sembra banale ma non lo è: chi ci dice che una transizione ci porti di nuovo ad uno stato a inoltro stabile?):
    - Supponiamo per assurdo che tali transizioni T' e T'' esistano
    - Poiché S è a inoltro stabile, S' dev'essere tale che $best_{S'}(i) = best_S(i)$ per ogni i=0,…,n (aiuto per le variabili: i in questo caso è un qualificatore)
    - Perciò, ci dev'essere (almeno) un nodo j per il quale $best_{S''}(j) \neq best_{S'}(j)$ (j è quindi l'identificatore del nodo per il quale cambia la rotta migliore)
    - Assumiamo che $best_{S'}(j) = S'(j,h)$ e $best_{S''}(j) = S''(j,k)$ (h e k sono rispettivamente i nodi per i quali passa la rotta migliore di j in S' e in S'')
    - Ricorda che:
        - $best_{S'}(h) = best_S(h) \Rightarrow j\ best_{S'}(h) = j\ best_S(h)$
        - $best_{S'}(k) = best_S(k) \Rightarrow j\ best_{S'}(k) = j\ best_S(k)$
        - $best_{S'}(j) = best_S(j) \Rightarrow S'(j,h) = S(j,h)$
        - S"(j,k) è più in alto in classifica di S"(j,h) (per il nodo j)
    - Abbiamo a questo punto tre possibilità (per far sì che la best per la riga j sia cambiata da h a k, per forza dev'essere successo che c'è un nuovo valore migliore sul k, un nuovo valore peggiore su h, o un nuovo valore su entrambi di cui il migliore è quello sul k):
        - $S''(j,h) = S'(j,h)$ e $S''(j,k) = j\ best_{S'}(k) \neq S'(j,k)$
        - $S''(j,h) = j\ best_{S'}(h) \neq S'(j,h)$ e $S''(j,k) = S'(j,k)$
        - $S''(j,h) = j\ best_{S'}(h) \neq S'(j,h)$ e $S''(j,k) = j\ best_{S'}(k) \neq S'(j,k)$
- Analisi dei casi
- **$S''(j,h) = S'(j,h)$ e $S''(j,k) = j\ bestS'(k) \neq S'(j,k)$**
    - $j\ best_{S'}(k) = j\ best_S(k)$ è più in alto in classifica di $S'(j,h) = S''(j,h)$
    - Perciò, esiste una transizione T*:S→S* tale che:
        - $S^*(m,n) = S'(m,n)$ for all $m \neq j$, $n \neq k$
        - $S^*(j,k) = j\ best_S(k)$
        - $best_{S^*}(j) \neq S^*(j,h)$ (perchè c'è almeno $j\ best_S(k)$ che è più in alto in classifica di $S'(j,h) = S^*(j,h)$)
        - $best_S(j) = S(j,h)$
    - Perciò, S non è in stato di inoltro stabile
- **$S''(j,h) = j\ best_{S'}(h) \neq S'(j,h)$ e $S''(j,k) = S'(j,k)$**
    - $S'(j,k) = S''(j,k)$ è più in alto in classifica di $j\ best_{S'}(h) = j\ best_S(h)$
    - perciò, esiste una transizione T*:S→S* tale che:
        - $S^*(m,n) = S'(m,n)$ pet tutti gli $m \neq j$, $n \neq h$
        - $S^*(j,h) = j\ best_S(h)$
        - $best_{S^*}(j) \neq S^*(j,h)$ (poichè c'è almeno $S'(j,k) = S^*(j,k)$ che è più in alto in classifica di $j\ best_S(h)$)
        - $best_S(j) = S(j,h)$
    - Perciò, S non è in stato di inoltro stabile
- **$S''(j,h) = j\ best_{S'}(h) \neq S'(j,h)$ e $S''(j,k) = j\ best_{S'}(k) \neq S'(j,k)$**
    - $J\ best_{S'}(k) = j\ best_S(k)$ è più in alto in classifica di $j\ best_{S'}(h) = j\ best_S(h)$
    - perciò, esiste una transizione T*:S→S* tale che:
        - $S^*(m,n) = S'(m,n)$ per tutti gli $m \neq j$, $n \neq h$, $n \neq k$
        - $S^*(j,h) = j\ best_S(h)$

        - $S^*(j,k) = j\ best_S(k)$
        - $best_{S^*}(j) \neq S^*(j,h)$ (poiché c'è almeno j $best_S(k)$ che è più in alto in classifica di j $best_S(h)$)
        - $best_S(j) = S(j,h)$
    - Perciò, S non è in stato di inoltro stabile

La dimostrazione di questo lemma non è banale, e non ci sono passaggi logici che saltano all'occhio in maniera immediata, e questo la rende molto macchinosa. Dopo qualche ora di studio approfondito ho capito sufficientemente la matematica che c'è dietro, ma è sicuramente difficile da imparare a memoria. In particolare il "perciò esiste una transizione T*" non è semplice da capire. In pratica la transizione T* è una copia della transizione T' "fabbricata in casa" da noi, in cui copiamo tutto da T' tranne quel dato che ci serve come artificio per la dimostrazione, e che quindi lo ricaviamo matematicamente. Nel caso in cui in S'' sulla riga j era cambiato h, in S* ci calcoliamo j,h. Nel caso in cui era cambiato k, in S* ci calcoliamo j,k. Nel caso di cambiamento di entrambi, ci calcoliamo entrambi. Grazie a questo artificio, calcolando quel valore e confrontandolo con le uguaglianze delle note, riusciamo a raggiungere l'assurdo.

Teorema

- Sia S uno stato ad inoltro stabile; non esiste alcuna sequenza finita di transizioni valide che porta il sistema ad uno stato S' tale che
    - $best_{S'}(i) \neq best_S(i)$ per qualche i=0,…,n
- Dimostrazione: facile! Supponiamo per assurdo che ci sia uno di questi stati intermedi. Applichiamo il lemma precedente e giungiamo subito ad un assurdo!

Uno stato di instradamento stabile è tale che, se il sistema è in quello stato, allora non può oscillare.
Osserva che l'esistenza di uno stato di instradamento stabile non implica:

- che il sistema lo raggiungerà (neanche dopo molto tempo)
- che tale stato di instradamento stabile sia l'unico

Osserva che la stabilità non implica l'ottimalità.
Inoltre, se uno stato è strettamente stabile, ciò non implica che il cammino p che va da i all'origine passando per j sia uguale sempre a i + il miglior cammino da j all'origine (questo più che altro perché quel percorso potrebbe non essere fra quelli permessi).

Il problema Stable Paths Problem (SPP – Problema dei Cammini Stabili)

Data un'istanza BGP, essa ammette (almeno) uno stato di instradamento stabile?
Qual è il tipo di stabilità più interessante?

Un'istanza di un problema SPP è semplicemente un'istanza del modello BGP costruito precedentemente. Potrebbe esserci una soluzione e potrebbe non esserci.

Bad Gadget: Quando l'istanza BGP del problema SPP non ammette soluzioni (o meglio, quando la soluzione è "no"), viene chiamata Bad Gadget. A un orale GdB ha chiesto anche di disegnare un esempio di Bad Gadget. Meglio prepararsene uno.
Attenzione alle politiche di backup. BGP non è robusto, cioè non garantisce recupero in caso di guasti. Uno stato stabile potrebbe diventare un Bad Gadget quando un si rompe un link.

Ci sono anche i cosiddetti equilibri precari, cioè quando per una parte dei nodi c'è una soluzione solo se un certo nodo non instrada su un certo cammino, ovvero quando una soluzione c'è, ma potrebbe essere "intrappolata" da una sfortunata serie di transizioni.

Un Modello per BGP – Raggiungibilità

Raggiungibilità

- Alcuni stati possono essere raggiunti attraverso una sequenza adeguata di transizioni (cioè, c'è un timing degli annunci che permette di raggiungerli)
    - L'annuncio seleziona i cammini solo quando "ne hai realmente bisogno"
- Ma non tutti i stati sono raggiungibili, anche quando partiamo da quello iniziale
- Uno stato S_2 è **irraggiungibile** da S_1 quando non c'è una sequenza di transizioni valide che porta da S_1 a S_2.

Aspetti Computazionali del Problema dei Cammini Stabili SPP

Teorema: SPP è NP-Completo

Dimostrazione
Spp appartiene a NP

- basta elencare tutti gli stati di instradamento
- e controllare la raggiungibilità di ognuno…

- Completezza
    - Riduciamo 3-sat a SPP
- 3-sat
    - Variabili: V = {X1, X2, …, Xn}
    - Clausole: disgiunzione di tre letterali (letterale = variabile singola normale o negata)
    - Domanda: c'è un'assegnazione di verità a V tale che ogni clausola sia vera?
- 3-SAT è NP-completo. Sapendo questo, per dimostrare che SPP è NP-Completo basta che dimostriamo che 3-SAT si può mappare a SPP (risolvere un problema che sappiamo essere sicuramente NP-completo risolvendo una sua riduzione, dimostra che tale riduzione è NP-completa, o almeno così pare)
- Da 3-SAT a SPP
    - Ogni variabile è associata con una struttura "disagree", di cui gli stati stabili rappresentano i valori di verità
    - Ogni clausola è associata ad un nodo
    - Viene usato un "bad gadget" per innescare l'instabilità *su richiesta* (nessuna soluzione per l'istanza 3-sat)

La riduzione non è proprio banalissima, anzi, è come al solito molto macchinosa. Ecco una buona spiegazione trovata sul forum.

“Dunque, vogliamo dimostrare che Stable Path Problem (un problema di decisione, cosi formulato: data una istanza BGP, ammette almeno un routing state che sia stabile?) sia un problema NP-Completo. Per far questo, riduciamo 3sat (che sappiamo essere NP-Completo e la cui formulazione sta in slide 40-1) a Spp e facciamo notare che trovare un algoritmo che risolvi Spp in tempo polinomiale equivarrebbe a trovare un algoritmo che risolva 3sat pure in tempo polinomiale (resta sottinteso il fatto che la riduzione deve svolgersi in tempo polinomiale).

La parte interessante sta nella fase di riduzione:

- ad ogni variabile di 3sat si associa una struttura disagree (la struttura disagree, con i suoi due stati stabili possibili, rappresenta proprio una variabile booleana: ad uno stato stabile associamo vero, all'altro associamo falso)
- ogni clausola (clausola = OR di al massimo 3 variabili, in 3sat) e' associata con un nodo
- ogni nodo (e quindi ogni clausola) e' associata con un bad gadget

La costruzione dell'istanza BGP relativa a una singola clausola si fa quindi nel modo seguente. Supponiamo di avere la clausola: X7 v -X5 v -X3:

- costruisco 3 strutture disagree
- costruisco un nodo C
- costruisco una struttura bad gadget
- collego il nodo C.1 di badgadget al nodo C, e collego il nodo C.0 al nodo 0
- collego C con i nodi delle strutture disagree che compaiono nella clausola

La costruzione completa la vedi in slide 41-2.
Affinche' sia una istanza BGP, mancano i path permessi, ordinati per ranking:

- le politiche delle strutture disagree sono note (un nodo, per raggiungere 0, preferisce sempre passare per il suo vicino piuttosto che raggiungerlo direttamente)
- le politiche del nodo C sono semplici: C raggiunge 0 passando solo per i nodi delle strutture disagree a cui e' collegato
- bad gadget e' bad gadget classico, con una novita': il nodo C.1 per raggiungere 0 preferisce passare per C quando cio' e' possibile (ovvero quando almeno una struttura disagree a cui e' collegato C deve consentire il passaggio diretto verso 0)

L'ultimo punto e' fondamentale: SE esiste un passaggio diretto di C verso 0 tramite una delle strutture disagree costruite, ALLORA C.1 usera' proprio quel path per raggiungere 0, disinnescando l'oscillazione di bad gadget.

Adesso arriviamo al dunque: SE siamo in grado di stabilire, in tempo polinomiale, se una rete siffatta ammette uno stato stabile o no, ALLORA siamo in grado di stabilire, in tempo polinomiale, se esiste una configurazione delle reti disagree che renda il sistema stabile. DIRE CHE esiste una configurazione delle reti disagree t.c. il sistema sia stabile EQUIVALE A DIRE CHE esiste una assegnazione degli stati delle reti disagree che rende il sistema

stabile. Ricordando che ad uno stato delle reti disagree corrisponde un valore di vero-falsita', "assegnare uno stato" vuol dire "assegnare un valore booleano" alle variabili in 3sat. Si potrebbe quindi riuscire a dire, in tempo polinomiale, se il problema 3sat di partenza ammette o no soluzione in tempo polinomiale."

La (in)stabilità di BGP – Conclusioni

Le questioni di stabilità sono molto interessanti.
Studiare la stabilità richiede di definire un modello.
Ci sono diverse nozioni di stabilità.
Data un'istanza BGP, se ne può cercare la stabilità (SPP).
Gli stati possono essere irraggiungibili.
Aspetti computazionali.

IPv6 – Indirizzamento

Header IPv6

Mentre l'indirizzo IPv4 è composto da 32 bit (4 byte) in notazione decimale (come 192.168.0.1/24), l'indirizzo IPv6 è composto da 128 bit (16 byte) in notazione esadecimale, ovvero 4 volte il numero di bit. La notazione prevede 8 blocchi da 16 bit ognuno:

es. 2001:0760:0002:0000:0000:5bbf:febc:5943/64

Se possibile la scrittura può essere semplificata: levando gli 0 a sinistra di un blocco, o raggruppando serie di 0:0:0 in un "doppio due punti" "::".

Es. 2001:760:2::5bbf:febc:5943/64

Le migliorie apportate da IPv6 rispetto a IPv4 sono molteplici.
C'è una semplificazione del formato dell'header per ridurre il carico computazionale sui router e contenere l'occupazione di banda. Il risultato è che l'header IPv6 è soltanto 2 volte più grande di quello IPv4. L'header IPv6 è quindi grande 40 byte.
Gli indirizzi IPv4 di sorgente e destinazione occupavano in tutto 8 byte dell'header, quindi il resto delle informazioni erano contenute in ben 12 byte.
Gli indirizzi IPv6, nonostante i 32 byte per indirizzi sorgente e destinazione, utilizzano solamente 8 byte per il resto delle informazioni.

C'è una nuova gestione delle opzioni per ottenere un inoltro più efficiente e garantire una maggiore estensibilità del protocollo. Il risultato è un consistente aumento delle prestazioni.

In IPv4:

- Version: 4 bit per specificare il formato dell'header del pacchetto IP. Per Internet Protocolo c'è il valore 4
- IHL: Internet Header Length, 4 bit per specificare la lunghezza dell'header del pacchetto IP in gruppi di 32 bit. Il valore minimo è 5 (20 byte: header IPv4 senza campo options)
- TOS: Type of Service, 8 bit per specificare i parametri del tipo di servizio richiesto. Usabile per definire la gestione del pacchetto durante il suo trasporto.
- Total Length: 16 bit per indicare la lunghezza totale del pacchetto
- Identification: 16 bit per identificare il frammento di un pacchetto nel caso esso sia frammentato
- Flags: 3 bit per controllare la frammentazione del pacchetto
- Fragment Offset: 13 bit per ordinare la ricostruzione di un pacchetto frammentato
- TTL: Time To Live, 8 bit, traccia il tempo di vita del pacchetto
- Protocol: 8 bit, specifica il successivo protocollo incapsulato di livello più alto
- Header Checksum: checksum dell'header IP incluse le opzioni
- Source IP address: indirizzo IP del mittente (32 bit)
- Destination IP address: indirizzo IP del destinatario (32 bit)
- Options: Lunghezza variabile
- Padding: Lunghezza variabile, serve per garantire che l'header del pacchetto sia allineato su 32 bit

L'header IPv6 è molto più semplice.

- Version: i soliti 4 bit per la versione. 6 per IPv6
- Traffic Class: 8 bit per identificare la priorità del pacchetto nel traffico Internet (simile al vecchio ToS). Una possibile applicazione può essere la differenziazione del traffico immesso in un ISP da un suo cliente. L'ISP può modificare questo campo per tutti i pacchetti in uscita verso altre reti per assegnare una classe di servizio concordata con altri ISP.
- Flow Label: 20 bit. L'utilizzo non è ancora chiaro. È usato per specificare uno speciale trattamento dei router fra la sorgente e la destinazione. Si dà uno stesso flusso (stesso valore di flow label) a pacchetti che hanno stesso indirizzo IPv6 sorgente e stesso indirizzo IPv6 destinazione. In questo modo si possono migliorare le prestazioni
- Payload Length: 16 bit. Specifica la lunghezza dei dati nel pacchetto (in byte, a quanto pare). Massimo pacchetti da 64 KB. Per dimensioni maggiori si utilizza Jumbo Payload.
- Next Header: Specifica l'header successivo. Se è un protocollo di livello più alto, i valori sono compatibili con quelli specificati per IPv4. Consente di specificare gli *extension header* (ma che cosa sono? Lo scopriremo tra poco).
- Hop Limit: 8 bit, è il vecchio TTL
- Source Address: 16 byte per l'indirizzo del mittente
- Destination Address: 16 byte per l'indirizzo del destinatario

Extension Headers

Quello degli Extension Header è un nuovo metodo per implementare le opzioni. In pratica invece di mettere un numero variabile di opzioni in un header di livello 3 che quindi può

diventare arbitrariamente variabile, le si mettono ciascuna in un header diverso, specificato nel campo Next Header dell'header precedente. Quindi ad esempio, se si vuole specificare qualche opzione di instradamento, nell'header IPv6 si mette il next header Routing nel campo apposito. Il livello 3 che aprirà il pacchetto troverà un payload incartato con l'header Routing (di grandezza uguale a quello IPv6 normale), che conterrà le informazioni per gestire il routing. Eventualmente scartando anche l'extension header Routing troverà il pacchetto TCP da passare al livello superiore. Insomma gli extension headers si comportano in tutto e per tutto come ogni altro header, solo che (almeno da quanto si capisce) vengono esaminati tutti dallo strato di rete della pila ISO-OSI.
I tipi di extension headers (e relativi codici per il campo Next Header) sono:

- 00 – Hop-by-hop options
- 43 – Routing
- 44 – Fragment
- 51 – Authentication
- 60 – Destination Options
- 50 – Encapsulating Security Payload
- XX – Protocolli di livello più alto come per IPv4
- 58 – Internet Control Message Protocol (ICMPv6)
- 59 – Nessun altro header

Hop-by-hop options
Informazioni che devono essere esaminate da ogni nodo lungo il percorso del pacchetto. Tra le opzioni utilizzate c'è *Router Alert* e *Jumbo Payload*.

Routing
Simile all'opzione IPv4 *Loose Source Route*, indica una lista di router da attraversare. Migliora le prestazioni rispetto a IPv4: l'header viene valutata solo dai router interessati. Ognuno di essi valuta il routing header e aggiorna la destinazione del pacchetto con l'indirizzo IPv6 del prossimo router della lista. È usato per il mobile IPv6 e multihoming.

Fragment
Usato solo dall'host mittente all'host destinatario ("I router non frammentano più!!!", come ricorda la dispensa).
IPv6 prevede una MTU di almeno 1280 byte. Link che non hanno questa capacità devono gestire la frammentazione e il riassemblaggio a livello data-link. Inoltre ogni nodo deve implementare una procedura di MTU Path Discovery (non strettamente necessaria).
Per inviare pacchetti più grandi della MTU consentita si utilizzano i fragment header.

Destination Options
Informazioni opzionali che saranno valutate solo dall'host destinatario. Può occupare 2 posizioni della "daisy chain" (tratto da wikipedia: "*Il modo più facile per aggiungere computer in una rete è detto "daisy-chaining": connettere ogni computer in serie con il successivo. Se un messaggio è destinato ad un computer su quella linea, ogni macchina lo passa alla successiva finché questo non raggiunge la sua destinazione.*"): prima del routing header e alla fine della catena. Viene usato nel mobile IPv6 insieme al routing header per risolvere il problema del cosiddetto routing "triangolare" (senza entrare nel dettaglio, permette ad un mobile di passare da una zona a un'altra rendendo il cambio di indirizzo IPv6 trasparente ai livelli superiori).

Tutte le implementazioni di IPv6 dovrebbero garantire il supporto alla sicurezza (IPsec), ma in realtà non è così.

Authentication Header
Fornisce l'autenticazione, un modo per controllare se l'indirizzo del mittente sia autentico e che il pacchetto non sia stato alterato durante il percorso.

Encapsulating Security Payload
Garantisce che solo il destinatario autorizzato sia in grado di leggere il pacchetto. Può agire in due modalità (come in IPv4): trasport o tunnel (che differenza c'è?).

L'ordine dei vari extension header non è arbitrario, ma dovrebbe essere il seguente:
IPv6 → Hop by hop (elaborate da ogni nodo) → Destination (elaborate da ogni router nella lista seguente) → Routing (lista dei router da attraversare) → Fragmentation (elaborate dal nodo destinazione) → Authentication (dal nodo destinazione dopo il riassemblamento) → Security (cifra tutto ciò che segue) → Destination (elaborate dal nodo destinazione) → Strato superiore

Gli Indirizzi IPv6

Se gli indirizzi IPv6 sono di 128 bit contro i 32 di IPv4 significa che il numero di indirizzi IP assegnabile in teoria è di gran lunga superiore (non quattro volte di più, quelli sono i bit!). Col massimo teorico ogni persona del pianeta potrebbe avere 10^{30} indirizzi. In realtà utilizzando la stessa efficienza di assegnazione di IPv4 avremmo in tutto una disponibilità di 10^{33} indirizzi IPv6.

Il Formato.
X:X:X:X:X:X:X:X
(8 campi da 16 bit in formato esadecimale). Il valore è indipendente dalla notazione delle lettere maiuscola o minuscola, gli zeri a sinistra di ogni campo possono essere omessi, campi successivi di zero sono rappresentati da **::** ma solo una volta in un indirizzo.
In una url gli indirizzi devono essere scritti tra parentesi quadre. È più scomodo per gli utenti (è più usato per scopi diagnostici), quindi è meglio usare una notazione per nome a dominio.

Gli indirizzi IPv6 si dividono in:
- Unicast: indirizzi di nodi
- Multicast: indirizzi di gruppi di nodi
- Anycast: indirizzi di servizi
  - Come gli indirizzi multicast identificano un gruppo di nodi ma diversamente dal multicast i pacchetti saranno consegnati al nodo tra essi più vicino al mittente (secondo le metriche dei router)

N.B. gli indirizzi broadcast sono stati eliminati, perché pericolosi riguardo ai DoS, *Denial of Service*, ovvero la tipologia di attacchi informatici dei cosiddetti *cracker* (chiamati anche *hacker*) volta a saturare una rete per negare l'erogazione di un servizio.

Una tabella sulla dispensa illustra l'architettura degli indirizzi IPv6. Ci sono alcune categorie riservate, alcune deprecate, notiamo che gli unicast a livello globale occupano 1/8 dello spazio di indirizzamento totale così come gli Unique Local Unicast. Poi ci sono i cosiddetti "scoped" (raggio d'azione), che sono i Link Local e i Site Local (deprecati) e gli indirizzi Multicast. Esaminiamo in dettaglio tutte queste categorie.

Unspecified

0:0:0:0:0:0:0:0 o semplicemente **::**
Indica l'assenza di indirizzo
Può essere usato nella richiesta iniziale DHCP per ottenere un indirizzo
Duplicate Address Detection (caratteristica introdotta con IPv6 che permette di rilevare indirizzi IP duplicati prima di assegnarne uno nuovo, al costo di un secondo di attesa di una risposta ad un messaggio multicast – DAD è anche disattivabile. Perché si trova sotto la voce Unspecified? Perché come detto poco fa l'indirizzo :: può essere usato per richiedere un nuovo indirizzo a DHCP)
Come 0.0.0.0/0 in IPv4, ::/0 indica la rotta di default.

Loopback

0:0:0:0:0:0:0:1 o semplicemente **::1**
Identifica il nodo stesso
Come 127.0.0.1 in IPv4 (localhost)
Per controllare se lo stack IPv6 funziona basta fare un ping6 ::1

IPv4 Compatible

Ora deprecati, erano usati in alcuni tipi di transizioni IPv4-IPv6. I primi 96 bit sono posti a 0 e gli altri specificano l'indirizzo IPv4
0:0:0:0:0:0:192.168.0.1
::192.168.0.1
::C0A8:1E01

IPv4 Mapped

Permettono di definire indirizzi IPv6 destinati a nodi che supportano solo IPv4.
I primi 80 bit sono posti a 0, i successivi 16 a 1 (FFFF) e gli ultimi 32 specificano l'indirizzo IPv4
0:0:0:0:0:FFFF:192.168.0.1
::FFFF:192.168.0.1
::FFFF:C0A8:1E01
Sono utilizzati per la transizione IPv4-IPv6. Perché quelli di prima sono deprecati e questi che sono la stessa cosa no? Probabilmente perché quelli di prima erano riservati a una certa tecnica di transizione, questi sono riservati ad un'altra tecnica, e quest'ultima ha prevalso.
Nota buffa: secondo la voce di wikipedia.it è IPv4 Mapped ad essere sconsigliato

I tipi di indirizzo visti finora sono assegnati al prefisso riservato 0000::/8

Subnet Prefix e Host Identifier

Gli indirizzi IPv6 si compongono principalmente di due parti:

Il prefisso di rete (primi 64 bit)
L'identificatore di interfaccia (Interface ID) (ultimi 64 bit)

L'host può essere identificato manualmente oppure viene calcolato come funzione dell'indirizzo MAC dell'interfaccia (identificati tramite EUI 64 – Extended Unique Identifier, tecnologia di assegnazione di identificatori agli indirizzi MAC con 64 bit)

L'interface ID:
- Identifica univocamente un'interfaccia
- Deve essere univoco su un link
- Può essere ricavato a partire dall'identificatore EUI-64

L'identificatore EUI-64 è in pratica l'evoluzione del MAC:
- Identifica il produttore (primi 3 blocchi da 8 bit) ed il "numero di serie" (ultimi 5 blocchi da 8 bit) di un'apparecchiatura con 64 bit

Esiste una procedura che consente di passare da EUI-48 ID (il vecchio indirizzo MAC) a EUI-64 ID

L'interface ID si ottiene dall'EUI-64 invertendo il settimo bit (chiamato Universal/Local).

Dalle dispense: "Quel settimo bit è complementato per semplificare la scrittura degli indirizzi locali. IPv6 infatti considera globali gli indirizzi col 7° bit dell'interface ID pari a 1.
Ad esempio *0:0:0:0*:0200::1 si può scrivere grazie al complemento a 1 del settimo bit come ::1."
(Non si capiscono alcune cose: Chi ti dice che complementando quel bit si semplifica l'indirizzo? Solo perché accade nell'esempio fatto non significa che accade sempre. Mettiamo ad esempio che come esempio diamo il risultato dell'esempio precedente… complementando, altro che semplificazione, facciamo l'operazione inversa. E poi perché dovrebbe essere sempre complementato? Se volessimo un IP globale? E poi anche ammesso che si semplificasse, perché tutta questa brama per una semplificazione talmente insignificante? Tra l'altro se semplifichi per il locale, non potrai goderne per l'universale, e viceversa).

Il vecchio indirizzo MAC a 48 bit utilizza per il "numero di serie" solo tre blocchi da 8 bit, quindi si aggiungono prima di essi 16 bit con valore FFEE per fare la traduzione.
Quindi ricapitolando: dal MAC address eventualmente si ricava la EUI-64. Dalla EUI-64 si complementa il settimo bit e si ricava l'interface ID, disponendo tutti i bit in notazione IPv6.

Nasce da qui un problema di privacy perché l'indirizzo MAC è associabile ad una persona specifica, e quindi dato che l'interface ID dipende dal MAC e quindi non cambia mai anche se cambia la rete, è possibile tracciare un utente. È comunque stato proposto un metodo alternativo per assegnare gli interface ID come stringhe casuali di 64 bit.

<u>Link e Site</u>
Un **link** è una rete fisica unica come ad esempio una LAN, un collegamento punto-punto o una rete geografica in tecnologia omogenea. Nodi sullo stesso link vengono detti *neighbor* (vicini).
Un **site** consiste invece in un gruppo di link gestiti da una stessa autorità (ad esempio un campus universitario).

Dal disegno si evince che un ISP non fa né link né site, e che AD un Provider vi si connettono dei site (che comprendono uno o più link).

Link Local

Tipica domanda d'esame: "quanti bit sono a zero?" -Palla 15/01/2009 15:07

- È un tipo di indirizzo “scoped” (ad ambito), una novità di IPv6.
- L’ambito in questo caso (lo scope) è un link locale (ad esempio una LAN o VLAN)
    - Un indirizzo Link Local può essere usato solo fra nodi dello stesso link
    - Non può essere ruotato (e che significa ruotare un indirizzo?)
- Fornisce ad ogni nodo un indirizzo IPv6 per iniziare le comunicazioni
- Viene configurato automaticamente su ogni interfaccia
    - Per farlo usa l’interface identifier basato sull’indirizzo MAC
- Il formato è FE80:0:0:0:<*interface identifier*>

Vedi le slide di ipv6-indirizzamento per vedere la grafica dei bit -Palla 15/01/2009 15:08
Non chiede tutti i prefissi degli indirizzi, ma bisogna conoscere in generale la struttura e la loro finalità (indirizzi link local, ecc.) -Palla 15/01/2009 15:10

Site Local

- È come Link Local solo con scope diverso: un intero site (ovvero una rete di link). Può essere dunque usato solo tra nodi dello stesso site.
- Non può essere usato fuori dal site (ad esempio in Internet)
- È molto simile agli indirizzi privati IPv4, ma è deprecato perché il concetto di site è risultato troppo vago e quindi è sostituito dagli indirizzi Unique Local.
- Nota: a me sembrava che anche i Link Local fossero simili agli indirizzi privati, chissà perché non è stato detto niente in proposito.

Unique Local

- Gli Unique Local sono simili ai site local, ma sono “quasi” globalmente univoci.
- Il formato è
    - FD:<global-id>:<subnet-id>:<interface-id>
        - FD sarebbe 00FD (8 bit)
        - Global-id è un numero casuale a 40 bit
        - Subnet-id è l’identificatore di sottorete a 16 bit (64.000 sottoreti indirizzabili)
- In realtà non sono globalmente univoci ma quasi, perché permettono interconnessioni di reti (a differenza dei 10.x.x.x di IPv4) e la probabilità di collisione è molto bassa:
    - Interconnettendo 1000 reti la probabilità è quasi un milionesimo.
- È vero però che non sono globalmente instradabili
    - Perché l’unicità non è comunque garantita
    - Non possono essere aggregati per via del global ID
- Quindi si userebbero solo per interconnettere reti molto grandi. A questo punto il dubbio è: ma non andavano bene i Link Local a ‘sto punto? La risposta credo sia no perché i Link Local hanno ambito su un Link, che è una semplice LAN, mentre gli Unique Local potrebbero essere utili in una grande azienda, usandoli al posto dei Site Local.

Global Unicast

- Sono gli indirizzi pubblici
- Iniziano con i tre bit 001 (Prefisso di Formato)
    - 2000:3FFF
- L'assegnazione di tali indirizzi segue una struttura gerarchica
    - IANA li assegna ai RIR (Regional Internet Registries, come RIPE NCC o ARIN)
    - I RIR li assegnano agli ISP
    - Gli ISP li assegnano ai site (tipicamente ai clienti)
    - I site vengono divisi in subnet
- Tale gerarchia, a differenza di IPv4, si riflette anche nei campi dell'indirizzo
    - IANA assegna da /19 a /23 ai RIR
    - I RIR assegnano delle /32 agli ISP
    - Gli ISP assegnano delle /48 ai customer
    - I customer suddividono in site in subnet /64
- La politica adottata dai RIR è di assegnare delle /32 agli ISP solo nei seguenti casi:
    - L'ISP prevede di assegnare almeno duecento /48 nei prossimi 2 anni
    - Ha un numero tale di clienti IPv4 da giustificare una /32
- Insomma gli indirizzi pubblici vanno assegnati al solito con la massima cautela

Multicast

- La comunicazione multicast significa "da uno a tanti"
- In IPv6 non esiste il broadcast. Al suo posto è usato Multicast, soprattutto nei link locali.
- Indirizzi ad ambito (scoped): sostituiscono il TTL di IPv4
- Formato:
    - FF<flags><scope>::<group id>
    - Identificati da FP (Format Prefix) 11111111 (=FF)
    - Flag = 0 permanente/1 temporaneo
    - Scope: node (1), link (2), site (5), organization (8), global (E)
    - Il Group ID identifica un gruppo multicast in un dato scope. Ovviamente per ogni scope gli identificativi dei gruppi devono essere univoci
- Esempio: Group-ID **All Nodes** (1)
    - Indirizzo FF02::1 → tutte le interfacce sullo stesso *link*
    - Indirizzo FF05::1 → tutte le interfacce sullo stesso *site*
    - Indirizzo FF0E::1 → tutte le interfacce su Internet
- Alcuni indirizzi multicast riservati sono "All Nodes" su link (FF02::1), "All Routers" su link (FF02::2), "All Routers" su site (FF05::2) e "Solicited Nodes" su link (FF02::1:FFxx:xxxx), che si usa nella Neighbor Discovery di ICMPv6 (verrà spiegato più avanti nel capitolo di ICMPv6).

Anycast

- Gli indirizzi anycast non sono distinguibili dagli unicast (non hanno uno specifico FP – Format Prefix)
    - Sono indirizzi unicast assegnati ad un insieme di interfacce, di solito di nodi diversi, a cui si dice esplicitamente che gli si sta assegnando un indirizzo anycast
- Indicano il server più vicino ad un mittente
- Alcuni di questi indirizzi sono riservati per usi specifici

- Router subnet
- Scoperta di home-agent per IPv6 mobile

Ogni host IPv6 deve riconoscere come propri i seguenti indirizzi:

- Un indirizzo *link-local* per ogni interfaccia
- Gli indirizzi *unicast-anycast* assegnati (manualmente o automaticamente)
- L'indirizzo di *loopback*
- L'indirizzo del gruppo *All-nodes multicast*
- Gli indirizzi *Solicited-node multicast* per ogni indirizzo *unicast/anycast* assegnato
- Gli indirizzi *multicast* di tutti gli altri gruppi di cui l'host fa parte

ICMPv6

ICMPv6 – Neighbor Discovery – Configurazione degli Indirizzi

ICMPv6 – Protocollo e Tipi di Pacchetto

Il protocollo ICMPv6 è l'equivalente IPv6 di ICMP, ovvero il protocollo di servizio che si occupa di trasmettere informazioni riguardanti malfunzionamenti, informazioni di controllo o messaggi fra i componenti di una rete.
Aggiunge inoltre nuove funzionalità
Neighbor Discovery (Scoperta dei Vicini)
Neighbor Solicitation, Irraggiungibilità, Autoconfigurazione
Fondamentalmente accorpa in un unico protocollo le funzioni svolte in IPv4 da ICMP (protocollo di servizio di livello rete per errori, controllo, diagnostica), ARP (protocollo di livello data link per conoscere il MAC del destinatario del pacchetto a partire dall'indirizzo IP) e IGMP (protocollo che in IPv4 gestiva i gruppi multicast).

<u>Formato dei Pacchetti</u>
Il codice di Next Header per il campo relativo dell'header di IPv6 per ICMPv6 è 58 che è diverso da ICMPv4.
L'header di ICMPv6 è formato da 4 campi:

ICMPv6 Type (Tipo) (8 bit)
ICMPv6 Code (Specifica ulteriore) (8 bit)
Header Checksum (intestazioni ICMPv6 e IPv6) (16 bit)
Dati ICMPv6

Il primo bit del campo Type distingue tra due classi di messaggi: i tipi da 0 a 127 sono segnalazioni di errori (error messages), quelli da 128 a 255 sono messaggi informativi (informational messages).

I messaggi di errore possono essere:

- Destination Unreachable
- Packet Too Big
- Time Exceeded
- Parameter Problem

I messaggi informativi possono essere di:

- **Diagnostica**
  - *Echo request ed echo reply*
- **Controllo**
  - Gestione dei gruppi multicast
    - *Multicast Listener Query/Report/Done*
  - Neighbor Discovery
    - *Router Solicitization/Advertisement*
    - *Neighbor Solicitization/Advertisement*
    - *Redirect*
    - *Inverse Neighbor Discovery*
- **Richiesta di informazioni**
  - *Node Information Query/Response*

ICMPv6 – Path MTU Discovery

Come già specificato in precedenza, in IPv6 la frammentazione è end-to-end, perché i router non frammentano più. Se ne occupa l'host sorgente. L'host deve sapere l'MTU del collegamento (Maximum Trasmission Unit, ovvero massima grandezza in byte del pacchetto ricevibile attraverso un protocollo di comunicazione). Per farlo utilizza la procedura di Path MTU Discovery (scoperta della MTU del cammino), basata sui messaggi ICMPv6 di tipo "Packet Too Big".
Questi messaggi sono generati dai router quando la linea su cui va inoltrato il pacchetto ha MTU inferiore alle dimensioni del pacchetto e riportano nel campo dati la MTU da utilizzare.

- Procedimento
  - Il nodo manda il primo pacchetto con una dimensione pari all'MTU del proprio link
  - Se riceve un messaggio d'errore "Packet Too Big", manda un nuovo pacchetto con le dimensioni indicate nel messaggio
  - Ripete questo procedimento finchè non riceve più messaggi di errore
- Il nodo manda periodicamente messaggi di dimensioni maggiori per rinnovare la stima
- In IPv6 l'MTU minima è di 1280 byte

Neighbor Discovery – Funzionalità di Base

La funzionalità ICMP di Neighbor Discovery è quella che rimpiazza l'ARP (Address Resolution Protocol, per trovare gli indirizzi MAC dati gli indirizzi IP).

Ancora, la procedura è fatta grazie a ICMPv6.
Fondamentalmente Neighbor Discovery gestisce le informazioni di controllo all'interno di un *link*.

- Risoluzione di Indirizzi
  - Neighbor Solicitation e Neighbor Advertisement
  - Neighbor Unreachability Detection
- Autoconfigurazione
  - Router Solicitation e Router Advertisement
- Redirect

I messaggi non possono uscire dal link e sono validi solo se hanno Hop Limit = 255.

Redirect
È simile al redirect ICMP di IPv4. Solo che in questo caso il redirect implica che il next hop (o direttamente la destinazione) si trova sullo stesso link. Infatti un router informa tramite ICMPv6 la sorgente che esiste un router migliore sul link per raggiungere la destinazione, o che vi esiste la destinazione stessa, restituendo nel messaggio l'indirizzo link local e il MAC del next hop o eventualmente della destinazione. La verifica che Next Hop sia pari a 255 riduce problemi di sicurezza che si riscontrano con IPv4.

Neighbor Solicitation e Advertisement
Equivalente ARP di IPv6. Usa pacchetti ICMPv6, è indipendente dal mezzo trasmissivo e può utilizzare i meccanismi di IPsec (IP security, suite di protocolli per internet orientati alla sicurezza).
Utilizza indirizzi multicast anziché broadcast, così da garantire una maggiore efficienza, poiché i nodi non interessati possono scartare il pacchetto già allo strato IP senza esaminarne il contenuto.
Per ottenere l'indirizzo fisico di un nodo destinazione, un nodo sorgente calcola l'indirizzo multicast Solicited-Node corrispondente all'indirizzo IPv6 del destinatario, dopodiché invia un messaggio di *Neighbor Solicitation* a questo indirizzo, e il nodo destinazione se presente lo riceve e gli risponde con un messaggio di *Neighbor Advertisement* in cui specifica nel campo dati il proprio indirizzo fisico. La sorgente mette questo indirizzo in una neighbor cache (equivalente della ARP cache).
Ovviamente questo procedimento è possibile perché ad ogni indirizzo ipv6 unicast corrisponde un indirizzo multicast di tipo Solicited Node formato aggiungendo gli ultimi 24 bit dell'indirizzo al prefisso ff02::1:ff00/104. In questo modo, in caso di interface ID formate da hardware si riducono le collisioni, e in caso di indirizzi multipli con lo stesso interface ID si riducono il numero di gruppi multicast a cui partecipare.

Neighbor Unreachability Detection (NUD)
Si tratta di un algoritmo che permette di individuare rapidamente guasti o cambiamenti di indirizzo fisico in maniera più efficiente di un semplice timeout. È utile per mobili che si spostano da un link ad un altro.
Ogni nodo tiene traccia dello stato di raggiungibilità dei propri vicini tramite informazioni provenienti da protocolli di strato superiore o inviando al nodo pacchetti di Neighbor Solicitation.
La procedura è semplice: se un nodo non ha informazioni sulla raggiungibilità di un vicino gli invia pacchetti **unicast** di Neighbor Solicitation, se non ottiene risposta cancella il vicino dalla cache (perché evidentemente il suo indirizzo fisico non è più quello) e rimanda un

Neighbor Solicitation, se questo fallisce vuol dire che il nodo è irraggiungibile. Le conseguenze dipendono dal tipo di nodo. Per gli host viene notificato un errore ai protocolli superiori, per i router il nodo ne seleziona un altro.

Neighbor Discovery – Autoconfigurazione degli Indirizzi

Autoconfigurazione senza stato.

IPv6 permette l'autoconfigurazione degli indirizzi automatica senza bisogno di un server DHCP. Infatti basta assegnare degli indirizzi link local in base all'Interface ID ottenuto tramite MAC, che è univoco a livello mondiale. I nodi possono comunicare tra loro utilizzando gli indirizzi link local. L'unica cosa che bisogna specificare a mano è il server DNS. Per fare tutto ciò c'è bisogno che il link supporti il multicast.

- Router Advertisement
- I router inviano periodicamente dei Router Advertisement per notificare a tutti gli host su un link (in multicast) molte informazioni utili
    - La presenza del router
        - Il suo indirizzo IPv6 link-local
        - Il suo indirizzo di livello 2
        - Disponibilità eventuale ad essere il default gateway
    - Parametri vari
        - Hop Limit, Reachable Time, MTU
    - La lista dei prefissi assegnati al link

Quando ricevono un router advertisement, gli host sul link ottengono indirizzi IPv6 semplicemente giustapponendo i prefissi ottenuti dal router al proprio Interface ID. In questo modo i nodi possono comunicare tra loro all'interno del link con gli indirizzi link-local, e attraverso i nodi fuori dal link con gli indirizzi globali ottenuti, grazie al router usando il suo link local come default gateway. Nota che ogni nodo ha un indirizzo globale per ogni prefisso comunicato dal router.

- Router Solicitation
- Un host appena avviato, per non attendere il successivo Router Advertisement, può semplicemente richiederlo.
- Basta fare un Router Solicitation, ovvero inviare tale messaggio ICMPv6 all'indirizzo Multicast che corrisponde a "tutti i router sul link". Ogni router risponde con un Router Advertisement con unicast soltanto al nodo che ha fatto la richiesta

Duplicate Address Detection

- Ogni volta che un nodo sta per assegnare un indirizzo ad un'interfaccia, il nodo verifica sempre che quell'indirizzo non sia già stato assegnato sullo stesso link (ciò non dovrebbe mai succedere se si utilizza l'autoconfigurazione senza stato).
- Funzionamento: basta che il nodo invii un Neighbor Solicitation all'indirizzo Solicited-Node corrispondente all'indirizzo che vorrebbe assegnare (chiamato *tentative address*, la sorgente di tale richiesta è invece l'unspecified address ::). Se non riceve alcuna risposta, l'indirizzo è valido e può essere assegnato.

Dunque, ricapitolando, un host all'avvio fa le seguenti cose:

- Configura gli indirizzi link-local
    - Generandone uno per ogni propria interfaccia
    - Verificando con Duplicate Address Detection che gli indirizzi siano unici sul link

- Utilizzando i link-local il nodo può comunicare con ogni altro nodo sul link a cui è connesso
- Se il nodo è un host, effettua ulteriori operazioni:
    - Emette dei Router Solicitation su tutte le interfacce
    - Per ogni Router Advertisement
        - Aggiunge il router alla lista dei router disponibili
        - Configura un indirizzo per ogni prefisso ricevuto nell'annuncio
    - Rimane in ascolto dei prossimi Router Advertisement

Ricordiamo ora alcune cose che è bene ricordare. I link local sono una cosa. Gli indirizzi globali sono un'altra. I link local possono essere ottenuti automaticamente e anche senza bisogno dei router. Gli indirizzi globali che si ottengono grazie alle comunicazioni dei prefissi da parte dei router sono invece degli Unique Local (o almeno credo, i global unicast ce li hanno solo le entità ai livelli superiori come ISP, RIR, e IANA). In pratica IPv6 concentra in sé tutte le tecniche IPv4 di aggregazione gerarchica e subnetting. Se ho l'indirizzo di un host da qualche parte nel mondo, sono sicuro che i primi tot bit saranno relativi al RIR, quelli dopo a un ISP, quelli dopo identificheranno una certa organizzazione, quelli ancora dopo una certa subnet, e gli ultimi identificheranno l'host, e quando il mio messaggio arriverà al router appartenente allo stesso link di quell'host, il router potrebbe certo operare una conversione di indirizzi per inoltrare il mio messaggio all'indirizzo link local della destinazione, ma potrebbe anche non farlo perché quell'host se tutto va bene dovrebbe considerare proprio anche l'indirizzo che ho usato io come destinatario, quindi a prima vista non ci sarebbe neanche bisogno di NAT. Queste sono solo considerazioni, ora riprendiamo il discorso.

Tempo di vita degli indirizzi
Nei Router Advertisement, ad ogni prefisso sono associati due tempi di vita in secondi

- Valid Lifetime: tempo per cui l'indirizzo può essere associato all'interfaccia
- Preferred Lifetime: tempo per cui l'indirizzo può essere utilizzato per le nuove connessioni

Se il primo vale 0, l'indirizzo è deprecato e non può essere utilizzato per nuove connessioni.
Se il secondo vale 0, l'indirizzo non è valido e non può rimanere associato all'interfaccia.
Gli host sono perennemente in ascolto dei Router Advertisement.
Intuitivamente il primo evento dice "questo indirizzo non è più valido", il secondo dice "non puoi creare nuove connessioni con questo indirizzo (ma quelle esistenti possono finire di usarlo)".
Questo meccanismo permette un semplice **renumbering** automatico degli host. Basta configurare un nuovo prefisso sul router e porre a zero il *Preferred Lifetime* di quello vecchio volutamente. L'identificazione del router sul link è fatta sempre secondo il suo link local e quindi non cambia. Gli host utilizzano nuovi indirizzi per nuove connessioni e i vecchi indirizzi (deprecati) per le connessioni esistenti. Alla fine del processo il vecchio prefisso viene rimosso dal router. Gli indirizzi sugli host vengono rimossi alla fine del *Valid Lifetime*.

Gestione Avanzata degli Indirizzi

<u>Router Renumbering</u>

- Se cambia il prefisso associato ad un site (o comunque ad un'organizzazione), devono essere riconfigurati tutti i prefissi dei router.
- Il protocollo di Router Renumbering semplifica questo processo
- Permette di cambiare indirizzi e prefissi dei router da una stazione di gestione
- Non affronta il problema del cambiamento dei record DNS
- Caratteristiche del protocollo
    - Riconfigura i router da remoto utilizzando opportuni messaggi di controllo originati da una stazione di gestione
    - Utilizza pacchetti ICMPv6 appositi
    - Il paradigma utilizzato è comando/risposta con ack e ritrasmissione
    - È obbligatorio l'uso di IPsec per autenticare i messaggi
- Attualmente questo protocollo non è largamente implementato

Configurazione Stateful

- Gli indirizzi e gli altri parametri di rete possono essere configurati anche manualmente. Sono possibili le seguenti alternative:
    - Configurazione interamente manuale
    - DHCPv6 (standard ancora in via di evoluzione)
    - Autoconfigurazione senza stato
        - I Router Advertisement contengono due flag che specificano le modalità di configurazione
            - "Managed Address Configuration" indica se l'host deve ottenere indirizzi anche da DHCPv6
            - "Other Stateful Configuration" indica se l'host deve utilizzare DHCPv6 per ottenere altre informazioni di configurazione, tipo DNS o NTP (Network Time Protocol, protocollo per sincronizzare l'orologio di sistema alla rete). Ovviamente sempre vero se è vera la prima

DHCPv6

- DHCPv6 viene utilizzato solo se sul link non ci sono router oppure se essi ne specificano l'utilizzo
- Gli indirizzi ottenuti si aggiungono a quelli ottenuti tramite autoconfigurazione
- Funzionamento
    - Simile a DHCP per IPv4
    - Il server mantiene informazioni sullo stato dei client
    - Permette di configurare gli indirizzi IPv6 o di gestire altre informazioni (tipo DNS o NTP)
    - Utilizza il protocollo UDP
    - Utilizza gli indirizzi multicast ff02::1:2 (tutti gli agenti DHCP con ambito link-local) e ff02::1:3 (tutti i server DHCP con ambito site-local)
- Rispetto all'autoconfigurazione senza stato, quella con DHCP:
    - Offre maggiore controllo sui singoli indirizzi
    - Offre maggiore sicurezza
        - Gli indirizzi non vengono assegnati a ciascun nodo sul link ma solo a quelli che soddisfano certe condizioni di sicurezza
    - Permette l'interazione col DNS
        - Tramite update dinamico

        - Tramite mappatura statica tra indirizzi di livello 2, indirizzi IP e nomi
    - Richiede la presenza di un server dedicato
- Lo standard è ancora in via di definizione
- Non è ancora molto implementato

Source Address Selection e Multihoming

Selezione dell'Indirizzo Sorgente

Un nodo ha in generale più indirizzi IPv6 (almeno un link-local per ogni interfaccia, più eventuali unique local, globali, ecc.). Quale utilizzare fra questi come sorgente (e destinazione) del flusso di dati? Il problema non è semplice. Tra l'altro un nodo potrebbe avere indirizzi IPv4 e IPv6, e indirizzi che non hanno validità globale.
Prima di scegliere l'indirizzo sorgente è bene scegliere l'indirizzo destinazione. Si immagini ad esempio un host a cui ci si vuole connettere, che ha un indirizzo IPv4 e un indirizzi IPv6. Se l'IPv6 è link-local e l'IPv4 è globale ha senso preferire il secondo, se c'è un IPv6 globale e un IPv4 autoconfigurato, meglio IPv6.

Algoritmo

- La primitiva *getaddrinfo()* restituisce una lista ordinata degli indirizzi (inclusi IPv4) di un host. Si ordinano in ordine di preferenza decrescente.
- Si provano gli indirizzi destinazione uno per volta, e per ognuno di essi, chiamiamolo D, si sceglie un indirizzo sorgente appropriato S = *Source(D)*
- L'algoritmo di scelta può essere soprascritto sia dall'amministratore che da un'applicazione
- *Source(D)* è scelto in base a regole precise (usare il giusto ambito, evitare indirizzi deprecati, preferire l'indirizzo più simile, cioè con cosiddetto *longest match*, eccetera)
- L'ordine di preferenza degli indirizzi destinazione è scelto in base a regole simili (evitare destinazioni irraggiungibili o deprecate, usare il giusto scope, longest match, eccetera)
- Alcune regole richiedono di calcolare *Source(D)*

Multihoming

In IPv4 il multihoming si basa su BGP. Questo fa nascere alcuni problemi.
Poniamo caso che un AS multihomed si interlacci via BGP a due ISP per motivi di load sharing e/o backup. Poniamo caso anche che gli indirizzi delle reti di tale AS siano *PI*, i cosiddetti indirizzi *Provider Independent*, chiamati così perché non sono originati dai Provider. In pratica né l'uno né l'altro ISP possono aggregare l'AS in downstream, e quindi i router degli ISP devono annunciare in upstream sia i prefissi originati dagli ISP stessi, sia i prefissi originati dal loro customer PI che fa multihoming, aumentando quindi le entry sulle tabelle di instradamento BGP. Un customer sarebbe aggregabile, e si direbbe *PA* (*Provider Aggregatable*) se originasse un indirizzo più specifico di quello del prefisso di uno dei suoi Provider. In questo caso l'entry sulle tabelle di instradamento dei router in upstream del Provider sarebbe una sola, quella più generale.

Insomma gli indirizzi PI non sono aggregabili e quindi sono una delle cause della crescita esponenziale delle tabelle di instradamento. Come si fa?
Notiamo che in IPv6 ogni interfaccia può avere più di un indirizzo, anche di Provider diversi. Una soluzione banale è questa: il customer non parla BGP.

- Soluzione banale: il customer non parla BGP
    - Non si aggiungono rotte alla tabella di routing
    - Le uniche entry nella default-free zone (ad esempio i router degli upstream provider) sono le /32 dei grossi Provider
- Problemi:
    - Non aggiunge ridondanza: se uno dei link va giù, metà delle connessioni muore
    - Non bilancia il traffico in ingresso: dipende da quale indirizzo viene scelto dai nodi del customer
    - Quindi se non c'è backup e non c'è load sharing… a che serve il multihoming in questo caso?

Il problema è molto serio, perché è impensabile utilizzare IPv6 per applicazioni critiche senza multihoming. Il problema è affrontato nel gruppo di lavoro "multi6" di IETF. Allo stato dell'arte sono stati studiati e documentati i requisiti, è stato scelto l'approccio da seguire ed è in corso la definizione dei dettagli del protocollo, che è stato chiamato **SHIM6**.

- Approccio scelto: *separa il localizzatore dall'identificatore*
- L'indirizzo IP svolge due funzioni:
    - Identifica il nodo ai protocolli di strato superiore
    - Identifica la posizione del nodo nella topologia della rete
- Queste due funzioni sono concettualmente diverse, quindi l'idea è di farle svolgere a indirizzi diversi
- Upper-Layer ID: identifica il nodo ai protocolli di strato superiore
    - Usato per gli scambi tra IPv6 e lo strato superiore. C'è un ULID per ogni nodo
- Locator: identifica la posizione del noto nella rete
    - Usati come indirizzi sorgenti/destinazioni dei pacchetti
    - Un nodo ha un locator per ogni indirizzo IPv6, e quindi uno per ogni prefisso di multihoming

ULID e Locator sono sintatticamente identici ma semanticamente diversi.
La connessione viene così aperta utilizzando l'ULID (indirizzo scelto con l'algoritmo di selezione dell'indirizzo sorgente). Poi se la connessione è persistente, e se peer lo permette, si inizializza shim6 (altrimenti no, per non evitare l'overhead di shim6) negoziando attraverso gli Extension Header. I due peer si scambiano "context tags" shim6, che identificano un contesto, cioè gli ULID della connessione e i locator attualmente in uso. Se c'è un problema i peer provano gli altri locator per ristabilire la connessione. Una volta trovata una coppia di locator funzionante, shim6 modifica tutti i pacchetti in transito cambiando traducendo da ULID a Locator in uso, comunicando i vari cambiamenti tramite extension header con context tag. Il protocollo di strato superiore continua a vedere una connessione fra gli ULID.
L'approccio è molto simile a NAT, con la differenza che non viola il principio della connessione end-to-end perché la traduzione è fatta dagli host (quindi si evitano tutti i problemi che NAT porta con sé).

Transizione IPv4 – Ipv6

Transizione IPv4-IPv6 – Il Processo di Transizione

IPv4 e IPv6, nonostante operino allo stesso livello della pila ISO-OSI e svolgano le stesse funzioni, sono di per sé protocolli incompatibili senza opportune tecnologie. È innegabile che il successo di IPv4 sia stato straordinario. IPv6 per potersi affermare deve garantire la compatibilità con i dispositivi IPv4 esistenti e fornire strumenti che facilitino il processo di transizione. I meccanismi devono essere essenzialmente semplici ed efficaci.
Tale necessità è stata riconosciuta fin nelle prime fasi dello sviluppo del nuovo protocollo, e buona parte dello sforzo progettuale è stata dedicata a tali meccanismi. Ne sono stati proposti numerosi, che variano a seconda dello scenario. La transizione sarà graduale, prevede una fase di coesistenza dei due protocolli e potrebbe durare decenni. Possiamo pensare ad un processo in tre fasi in cui nella prima la rete IPv6 si appoggia all'infrastruttura IPv4, nella seconda i due protocolli coesistono e nell'ultima è la rete IPv4 ad appoggiarsi su IPv6. Comunque alla fine le applicazioni dovranno supportare tutte IPv6.
I meccanismi di compatibilità si classificano in tre categorie fondamentali:

- Implementati sugli host
    - Host Dual Stack
    - Altri: BIS, BIA (anche se questi rientrano anche nei traduttori di protocollo)
- Implementati a livello di rete
    - Tunnel
        - Manuali, 6to4, automatici
        - Altri: ISATAP, Teredo
    - Rete Dual Stack
- Basati su traduttori di protocollo
    - SIIT e NAT-PT
    - Altri: TRT, …

Host Dual Stack
È un approccio molto semplice: in pratica un nodo dual stack implementa sia l'uno che l'altro protocollo, e ha entrambi gli indirizzi anche sulla stessa interfaccia. Si chiama Dual Stack perché la pila ISO-OSI ha due livelli di rete. Per le applicazioni che usano solo IPv4 usano IPv4 altrimenti eventualmente usano IPv6. È sì molto semplice ed efficiente ma in pratica non si fa nulla per ridurre il fabbisogno di indirizzi IPv4: è più un meccanismo di compatibilità che di transizione. Attualmente quasi tutti i nodi IPv6 sono dual stack.

Meccanismi di Compatibilità a Livello di Rete

Tunnel IPv6-in-IPv4

I tunnel sono essenzialmente dei meccanismi che servono per trasportare un protocollo in una rete basata su un altro protocollo. I tunnel IPv6-in-IPv4 permettono di utilizzare IPv6 senza predisporre di un'infrastruttura di rete nativa IPv6. I pacchetti IPv6 vengono incapsulati in pacchetti IPv4 semplicemente aggiungendo un header IPv4 (con campo protocol 41 per IPv6).
Gli estremi del tunnel sono cosiddetti Router dual-stack. All'ingresso del tunnel sono loro che incapsulano i pacchetti IPv6 in quelli IPv4 e, dopo che essi sono stati instradati normalmente lungo la rete IPv4 come se fossero IPv4 vengono decapsulati ancora dal router dual-stack che invia il vero pacchetto IPv6 alla destinazione.
Da fuori il tunnel appare come un solo hop IPv6. Dal punto di vista di IPv6 la rete IPv4 è come se fosse una tecnologia trasmissiva di livello 2 (difatti il pacchetto IPv6, che essendo un pacchetto di rete di solito dovrebbe venire incapsulato da un header data-link, viene invece incapsulato dall'header IPv4, che però per lui è come se fosse l'atteso data link).
La MTU di un tunnel si riduce di 20 byte a causa della presenza header IPv4 (ad esempio, poniamo che il nodo voglia inoltrare un pacchetto di 1400 byte, e che la rete IPv4 supporti una MTU di 1430 byte. Bisogna calcolare che il livello data link non incapsulerà solo i 1400 byte del pacchetto IPv6 ma anche i 20 byte dell'header IPv4, quindi la MTU a tal fine è come se fosse di 1410 byte).
Le topologie possibili dei tunnel sono router verso router, host verso router e host verso host. Queste tecnologie sono importanti per le prime esperienze con IPv6 e nella prima fase di transizione.

- I tunnel possono essere creati configurando a mano gli estremi, al fine di stabilire un collegamento punto-punto permanente tra due router dual stack (e definire così un tunnel). I tunnel **configurati** sono ampiamente utilizzati.
- I **Tunnel Broker** sono delle applicazioni web raggiungibili tramite IPv4 alle quali dei client richiedono la formazione di un tunnel tramite una pagina web. Il broker identifica l'utente e configura un router per creare il tunnel fino all'utente, comunicando a quest'ultimo i parametri. A questo punto l'utente può sfruttare il tunnel appena creato per arrivare a nodi IPv6. In pratica i tunnel broker creano tunnel "su richiesta". Sono molto utilizzati per utenti "occasionali".
- I **tunnel automatici** possono essere creati facendo corrispondere il destinatario del pacchetto ad uno degli estremi del tunnel. Usando gli indirizzi IPv4-compatibili si determina l'indirizzo IPv4 dall'indirizzo destinazione. Ma non riducono il fabbisogno di indirizzi IP e nel caso in cui il mittente sia un nodo dual stack tanto vale utilizzare IPv4.

<u>Tunnel 6to4</u>
Permettono di connettere interi site IPv6, dando loro un indirizzo IPv4 pubblico che funge da estremo del tunel. La rete ha prefisso 2002:aabb:ccdd::/48 dove il 2002 è il prefisso 6to4 assegnato da IANA e aabb:ccdd è la rappresentazione esadecimale dell'indirizzo IPv4.
In pratica in questo modo un site IPv6 è visto dalla rete come un unico nodo IPv4 con relativo indirizzo. La rete all'interno del site ha quel prefisso particolare che identifica le reti IPv6 con quel tipo di indirizzo riservato.
Un indirizzo 6to4 è un indirizzo IPv6 che si trova all'interno di un site con prefisso riservato 2002:eccetera. Un pacchetto destinato a un indrizzo 6to4 viene riconosciuto, inoltrato fino al router 6to4, che ne deduce l'estremo del tunnel, lo incapsula e lo instrada su IPv4 facendolo viaggiare finchè non arriva all'altro estremo del tunnel che lo decapsula e lo inoltra sul site di arrivo.

In questa forma però c'è l'inconveniente che i nodi su reti 6to4 non possono inviare pacchetti agli altri nodi del resto della rete IPv6 (anche se nodi IPv6 generici potrebbero inviare pacchetti a nodi 6to4: basta che ci sia un router 6to4 a cui far arrivare questi pacchetti). Quindi la comunicazione col resto della rete IPv6 sarebbe impossibile.
La soluzione si incarna nei cosiddetti **6to4 Relay Router**. Questo tipo di router è disposto a fornire un accesso alla rete IPv6 inoltrando pacchetti provenienti da nodi 6to4. Viene utilizzato in una sola direzione (facendo così un percorso asimmetrico) e impiega la banda Internet di chi lo mette a disposizione.
Esistono vari tipi di Relay Router pubblici. Per non doverlo configurare manualmente è stato messo a disposizione un indirizzo anycast per i Relay Router (così da trovare direttamente il Relay Router più vicino). Tale indirizzo è 2002:c058:6301:: corrispondente all'indirizzo IPv4 192.88.99.1.
Grazie ai Relay Router, il tunneling 6to4 diventa un meccanismo di transizione efficace ed accessibile.
Note: le politiche di transito sulla rete IPv4 porranno fine prima o poi ai Relay Router pubblici, ed i provider IPv4 potrebbero iniziare a fornire dei Relay Router ai loro clienti.
Ricapitolo brevemente l'uso dei Relay Router. Un host su un site 6to4 vuole mandare un pacchetto a un host situato su un site IPv6 nativo e quindi l'indirizzo sarà *blablablabla* che non corrisponde affatto ad un indirizzo 6to4 perché è tipo un unicast IPv6 nativo ad esempio. Comunque il pacchetto va a finire sul router 6to4 del site 6to4 perché è l'unico modo che il site ha per farlo uscire sulla rete. A questo punto il router 6to4 si chiede "e ora come lo converto questo? Non è qualcosa che posso trasformare in un indirizzo IP vecchio modello" e allora che fa? Lo incapsula mandandolo a 192.88.99.1. Ma chi è questo tizio? È il Relay Router più vicino. Come gli arriva questa informazione al router 6to4 da una rete IPv4? Per il router la rete IPv4 non è altro che una tecnologia di livello 2 che gli incapsula i pacchetti IPv6, che all'inizio conterranno apposite informazioni di controllo per sapere chi è questo Relay Router più vicino. Finalmente gli manderà il pacchetto che arriverà al Relay Router. Il Relay Router ha un'interfaccia che comunica IPv6 nativo e quindi potrà far arrivare il pacchetto al vero destinatario *blablablabla*. Quest'ultimo, dato che i pacchetti da IPv6 nativo a 6to4 invece si possono mandare benissimo tramite un apposito router 6to4 situato nell'IPv6 nativo, glieli manda per di là. I percorsi di andata e di ritorno sono dunque asimmetrici.
I vantaggi di questo tipo di tecnica sono che è semplice da configurare, permette di utilizzare IPv6 senza disporre di indirizzi (grazie agli indirizzi riservati 2002eccetera) e senza avere un provider IPv6 nativo e non richiede alcun supporto particolare ai nodi (i prefissi 6to4 possono essere appresi tramite il solito meccanismo di autoconfigurazione stateless).
Gli svantaggi sono che gli indirizzi IPv6 sono legati all'indirizzo IPv4 del router di bordo. Se questo indirizzo cambia bisogna fare un renumbering di tutti gli indirizzi interni al site. Inoltre il Relay Router può essere molto lontano sia dal nodo sorgente in IPv4 che dal nodo destinazione in IPv6.

Altri tipi di tunnel
non sono nel programma del corso di impianti -Palla 23/01/2009 09:57

- ISATAP
  - "Intra-Site Automatic Tunnel Addressing Protocol"
  - Ha lo scopo di connettere tra loro nodi e router IPv6 su una rete di un'organizzazione "IPv4-only"
- Teredo
  - "Tunneling IPv6 over UDP Through NATs"

        - Incapsula i pacchetti IPv6 in pacchetti UDP (di IPv4) anziché direttamente in pacchetti IPv4
        - Permette l'utilizzo di tunnel anche in presenza di NAT IPv4
            - Utilizzando tecniche non banali di "NAT traversal"
        - Sviluppato dalla Microsoft
            - Incluso nelle prossime versioni di Windows
            - Proposto anche come soluzione generale per il NAT traversal
                - È più semplice utilizzare le API IPv6 che riscrivere NAT traversal da capo

Reti Dual Stack

- I tunnel sono stati molto usati nella fase iniziale ma andranno a diminuire per i seguenti motivi:
    - Una rete di tunnel è difficile da gestire
    - Le prestazioni sono inferiori a quelle di una rete nativa
    - I tunnel rendono la rete IPv6 dipendente dalla rete IPv4
- Migliore soluzione: **reti dual stack**
    - Backbone Dual Stack: stessa topologia per IPv4 e IPv6
        - Minori difficoltà di gestione e costi
        - Non richiede l'utilizzo di molti IPv4
    - Accesso a seconda degli utenti
        - Il NAT potrebbe essere sostituito da IPv6 nativo con NAT-PT (di cui vediamo tra poco)
    - Una rete nativa capillare favorirà la diffusione di IPv6

Traduzione di Protocollo

La traduzione di protocollo è l'unico modo per far parlare IPv4-only con IPv6-only. Può essere un'alternativa ai nodi Dual Stack. È importante sapere dove posizionare tali traduttori. Si noti che tutto il traffico passerà da tali traduttori e ciò può causare problemi di sicurezza, robustezza, traffico, eccetera.
In molti meccanismi di questo tipo gli indirizzi IPv4 sono rappresentati come particolari indirizzi IPv6 (dato che lo spazio di indirizzamento è più ampio è possibile). Gli indirizzi IPv4 rappresentati in questo modo sono inviati verso il traduttore.
Le traduzioni di protocollo possono essere implementate:

- A livello IP
    - SIIT, NAT-PT
- A livello di trasporto
    - TRT
- Modificando la pila protocollare degli host
    - BIS, BIA

SIIT

- Stateless IP/ICMP Translation Algorithm
- Meccanismo generale che permette ai nodi IPv4 e IPv6 di comunicare attraverso un traduttore

- Funzionamento:
- Da nodo IPv6 S a nodo IPv4 Q: S può mandare solo header IPv6. Come sorgente mette il proprio indirizzo IPv6? No: il nodo S ottiene un indirizzo IPv4 temporaneo che gli dà qualche dispositivo (tipo DHCP), da questo si crea un indirizzo IPv6 di tipo IPv4-translated da usare come sorgente. E come destinatario S cosa usa? Lui conosce l'IPv4 del destinatario ma non può usarlo. Al suo posto usa la mappatura di tale indirizzo in IPv4-mapped. Quindi invia il pacchetto. Q può ricevere solo header IPv4, quindi il pacchetto deve passare dal traduttore, che decapsula l'header IPv6 e vi mette al suo posto l'header IPv4. La traduzione degli indirizzi è semplicissima: la sorgente è un IPv4-transalted e il destinatario è un IPv4-mapped, quindi basta usare i corrispettivi indirizzi IP. Si intuisce come anche sulla strada di ritorno la traduzione sia banale.

I vantaggi sono che il nodo traduttore non deve tener traccia delle connessioni. L'architettura è scalabile e robusta. Gli svantaggi sono che richiede modifiche alle implementazioni IPv6, la presenza di un meccanismo di assegnazione degli indirizzi IPv4 temporanei e altri problemi di gestione.

NAT-PT

Il NAT-PT fonde le tecniche del NAT con quelle di SIIT.
In pratica si immagini il traduttore NAT-PT che si interpone fra due nodi IPv6 only (S) e IPv4 only (Q). Questo NAT-PT dispone anche di un meccanismo detto di DNS ALG, e ora spiegheremo a cosa serve.
Il NAT-PT dispone di due prefissi: un prefisso IPv6 che usa dalla parte della rete IPv6 per dare un rappresentante IPv6 al nodo S, e un pool di indirizzi IPv4 che usa dalla parte della rete IPv4 per dare un rappresentante IPv4 al nodo Q.

- S vuole inviare un pacchetto a Q e sa il suo nome, ad esempio www.qualcosa.com. Fa una richiesta di name lookup per quel nome, e il DNS ALG nel NAT-PT intercetta la query. Allora prima richiede al DNS l'indirizzo IP corrispondente, poi associa, attraverso una funzione deterministica, a quell'indirizzo IP un indirizzo IPv6 (con prefisso che è quello associato) da restituire al nodo S, che lo vedrà come nodo destinatario (in realtà è un proxy usato dal NAT-PT). Nota: non serve che si memorizzi in una tabella questa associazione <indirizzo IPv4 vero – indirizzo IPv6 proxy> perché si ottiene deterministicamente e quindi sarà sempre lo stesso. Il NAT-PT invece associa all'indirizzo dell'host IPv6 S un indirizzo IPv4 preso dal pool (in maniera dinamica), e mettendolo nella tabella tipo NAT classico per tenerne traccia. L'indirizzo IPv4 scelto sarà a sua volta il proxy che fungerà da sorgente per il nodo ricevente che si trova dall'altra parte sulla rete IPv4.
- Quindi quando il pacchetto arriva sul traduttore NAT-PT, cosa succede? Il traduttore grazie al meccanismo dei DNS sa l'IPv4 destinatario qual è. Allora dà un nodo destinatario fittizio IPv6 alla sorgente che è funzione deterministica dell'IPv4 reale. S crederà di parlare con un destinatario IPv6 che è questo qui fittizio. Nel frattempo il traduttore estrapola un indirizzo IPv4 fittizio da creare come proxy dalla parte del ricevitore, per far credere al destinatario che sia un IPv4 a parlargli. Crea quindi una entry in tabella <indirizzo IPv6 di S – indirizzo IPv4 fittizio>. A questo punto quando arriva il pacchetto il traduttore che fa? Traduce la sorgente IPv6 con la corrispettiva voce IPv4 fittizia nella tabella, e la traduzione del mittente è fatta. La traduzione del destinatario è semplice perché abbiamo detto che il NAT-PT sa qual è l'IPv4 del destinatario e quindi ci mette quello.

- A ritorno invece che succede? Il pacchetto che arriverà sarà con header IPv4, sorgente indrizizzo IPv4 vero di Q e destinatario indirizzo IPv4 fittizio del NAT-PT. Il traduttore come sorgente ci mette l'indirizzo IPv6 fittizio ricavato di nuovo come prima (tanto è deterministico quindi esce sempre lo stesso), e la destinazione la ricava sempre dalla entry della tabella.
- Vantaggi: trasparente ai nodi che lo utilizzano
- Svantaggi: stessi problemi del vecchio NAT: fragilità, necessità di algoritmi specifici per gestire protocolli che non sono end-to-end e non permette connettività diretta da estremo a estremo. C'è da dire però che questi svantaggi non hanno certo impedito la diffusione di NAT IPv4, e ci sono già molte applicazioni che supportano NAT.
- NAT-PT non è ancora largamente implementato.

Transport Relay Router
È una tecnica di traduzione di protocollo che lavora a livello di trasporto. Un nodo detto Relay (tramite) si prende la briga di instaurare due connessioni. Il nodo IPv6 crede di aver instaurato una connessione TCP col nodo destinazione. Il nodo IPv4 crede di aver instaurato una connessione TCP col nodo sorgente. In realtà è il Relay che ha instaurato due connessioni.
Il funzionamento è molto riassuntivo, comunque è ovvio che sono necessarie modifiche alle risoluzioni di nomi o indirizzi e di una memoria per le informazioni di stato.

BIS e BIA
- Metodi di transizione implementati via software
- Bump-In-the-Stack (BIS)
    - Permette a nodi IPv4 di comunicare con nodi IPv6 senza traduttori esterni
    - È uno strato software inserito tra lo stack IP e il livello data-link
    - Effettua la traduzione SIIT internamente al nodo
- Bump-In-the-API (BIA)
    - Permette di utilizzare IPv6 con applicazioni che non lo supportano
    - Traduce le chiamate alla socket API IPv4 a chiamate alla socket API IPv6
- Questi metodi non sono molto implementati

Netkit e IPv6 – Comandi Base

Progetto Netkit: *"`The poor man's system to experiment computer networking`"*.

Comandi per i file di configurazione

I questa sezione vengono elencati i principali comandi per la gestione dei file di configurazione dei laboratori di netkit

Perering configuration commands

!<comment-on-a-single-line>
router bgp <as-number>
neighbor <ip-neighbor> remote-as <neighbor-as-number>
neighbor <ip-neighbor> description <text>

Annuncement commands

network <network-ip> mask <network-mask>
network <network-ip/network-mask>

Prefix filtering commands

neighbor <ip-neighbor> prefix <p-list-name> in
neighbor <ip-neighbor> prefix <p-list-name> out

ip prefix-list <p-list-name> permit <network/mask>
ip prefix-list <p-list-name> deny <network/mask>

Attribute setting commands

neighbor <ip-neighbor> route-map <route-map-name> in
neighbor <ip-neighbor> route-map <route-map-name> out

route-map <route-map-name> permit <sep-number>
match <annuncce-propriety>
set <attribute-setting>
...

route-map <route-map-name> deny <sep-number>
match <annuncce-propriety>
set <attribute-setting>
...

Comandi per le componenti di rete (da fare!!!)

Il Sistema Pubblico di Connettività

Da Wikipedia

Il Sistema Pubblico di Connettività (SPC) è la rete che collega tra loro tutte le amministrazioni pubbliche italiane, consentendo loro di condividere e scambiare dati e risorse informative.

Istituito e disciplinato dal Decreto legislativo del 28 febbraio 2005, n. 42, esso viene definito come "l'insieme di infrastrutture tecnologiche e di regole tecniche per lo sviluppo, la condivisione, l'integrazione e la diffusione del patrimonio informativo e dei dati della pubblica amministrazione, necessarie per assicurare l'interoperabilità di base ed evoluta e la cooperazione applicativa dei sistemi informatici e dei flussi informativi, garantendo la sicurezza, la riservatezza delle informazioni, nonché la salvaguardia e l'autonomia del patrimonio informativo di ciascuna pubblica amministrazione".

Il Sistema Pubblico di Connettività - indicato a volte come Sistema Pubblico di Connettività e Cooperazione - viene gestito dal Centro Nazionale per l'Informatica nella Pubblica Amministrazione (CNIPA).

Principi di base:

1. Sviluppo architetturale ed organizzativo atto a garantire la natura federata, policentrica e non gerarchica del sistema.
2. Economicità nell'utilizzo dei servizi di rete, di interoperabilità e di supporto alla cooperazione applicativa.
3. Sviluppo del mercato e della concorrenza nel settore delle tecnologie dell'informazione e della comunicazione.

Obiettivi:

- Fornire un insieme di servizi di connettività condivisi dalle Pubbliche Amministrazioni (PA) interconnesse, graduabili in modo da poter soddisfare le differenti esigenze.
- Garantire l'interazione della PA centrale e locale con tutti gli altri soggetti connessi a internet, nonché con le reti di altri enti, promuovendo l'erogazione di servizi di qualità per cittadini e imprese.
- Fornire un'infrastruttura condivisa di interscambio che consenta l'interoperabilità tra tutte le reti delle PA esistenti.
- Fornire servizi di connettività e cooperazione alle PA che ne facciano richiesta, per permettere l'interconnessione delle proprie sedi e realizzare cosi' anche l'infrastruttura interna di comunicazione.
- Realizzare un modello di fornitura dei servizi multifornitore coerente con l'attuale situazione di mercato e le dimensioni del progetto stesso.
- Garantire lo sviluppo dei sistemi informatici nell'ambito del SPC salvaguardando la sicurezza dei dati, la riservatezza delle informazioni, nel rispetto dell'autonomia del patrimonio informativo delle singole amministrazioni.