IPv4-in-IPv6: una nuova strategia per la transizione a IPv6



Massimo Vellucci[2], Massimo Bernaschi[1], Luca Vollero[2]

[1] *CNR-IAC, Istituto per la Applicazioni del Calcolo*
[2] *Università Campus Bio-Medico di Roma*

Abstract. IPv4 è il protocollo di comunicazione adottato dalla maggioranza dei dispositivi collegati a Internet. Tuttavia, essendo un protocollo definito nelle prime fasi dello sviluppo di Internet, il suo utilizzo pone problemi legati all'esaurimento degli indirizzi IPv4 disponibili e ad altre sue limitazioni costitutive. IPv6 è la nuova versione del protocollo Internet, definita con l'obiettivo di superare i limiti di IPv4.
La sfida principale nella diffusione di IPv6 è data dalla disponibilità di meccanismi efficaci per l'integrazione dei servizi attivi in reti IPv4 su reti IPv6, minimizzando i costi ed i rischi connessi alla migrazione.
Il presente lavoro propone un nuovo meccanismo basato sul tunneling dinamico di pacchetti IPv4 in IPv6 e denominato, per questo, IPv4-in-IPv6. L'approccio proposto è rivolto agli operatori di rete che vogliono semplificare la configurazione e la gestione del loro backbone IPv6, mantenendo il pieno supporto di IPv4 sulle reti di accesso.

1. Introduzione

IPv4, (1) è il protocollo di comunicazione adottato dalla maggioranza dei dispositivi collegati a Internet. Tuttavia, essendo un protocollo definito nelle prime fasi dello sviluppo di Internet, il suo utilizzo pone dei problemi legati all'esaurimento degli indirizzi IPv4 disponibili e ad altre sue limitazioni costitutive in ambito di mobilità e sicurezza. IPv6, (2), detto anche IP next generation (IPng), è la nuova versione del protocollo Internet, definita con l'obiettivo di superare i limiti di IPv4. Il problema principale nella migrazione a IPv6 è legato al gran numero di applicazioni e servizi Internet basati su IPv4, che non sopravvivrebbero a una banale sostituzione di protocollo. La sfida principale nella diffusione di IPv6, dunque, si traduce nella disponibilità di meccanismi efficaci per l'integrazione dei servizi attualmente attivi in reti IPv4 su reti IPv6 (3), minimizzando i costi ed i rischi connessi alla migrazione (4).

Con lo scopo di facilitare una graduale integrazione tra IPv4 e IPv6, il Working Group IPng Transition (ngtrans) della IETF (Internet Engineering Task Force) ha analizzato le questioni legate a tale transizione (5) e ha proposto diversi meccanismi, che costituiscono un "transition toolbox" (6), volti a facilitare l'interoperabilità tra i due protocolli e la migrazione a IPv6. Il presente lavoro propone un nuovo meccanismo basato sul tunneling dinamico di pacchetti IPv4 in IPv6 e denominato, per questo, IPv4-in-IPv6. Tale meccanismo si aggiunge a quelli disponibili nel suddetto "transition toolbox". L'approccio proposto è rivolto agli operatori di rete che vogliono semplificare la configurazione e la gestione del loro backbone IPv6, mantenendo il pieno supporto di IPv4 sulle reti di accesso e per i data center che necessitano di un ritardo per l'introduzione del nuovo protocollo

2. La soluzione IPv4-in-IPv6

2.1 Stato dell'arte nelle soluzioni di tunneling IPv4 over IPv6

La maggior parte delle soluzioni di tunneling prevede la gestione di flussi IPv6 incapsulati in tunnel IPv4 (6) (7) (8) (9) (10) (11) (12), e fa riferimento a scenari in cui la transizione è all'inizio e si ha la necessità di collegare isole IPv6 per mezzo di un'infrastruttura IPv4. Più recentemente, sono state proposte diverse soluzioni che si basano sul tunneling opposto, con lo scopo di gestire scenari in cui IPv6 è il protocollo dominante nell'infrastruttura di rete. Queste soluzioni si basano prevalentemente sulle specifiche di tunneling su IPv6 definite in (13). L'aspetto principale che differenzia questi meccanismi di tunneling è la configurazione del tunnel IPv4-over-IPv6 e la gestione del processo

Workshop GARR 2014 - Next Network: Costruiamo il futuro della rete - Selected Papers. Roma 2-4 dicembre 2014
ISBN: 978-88-905077-5-5 *© 2016 Associazione Consortium GARR*

di incapsulamento:

- 4in6 (13) è l'approccio più semplice e si basa sulla configurazione manuale di tunnel statici combinati con una conversione di indirizzo stateless. 4in6 può essere combinato con il Tunnel Setup Protocol (TSP) (14) al fine di facilitare la configurazione dei tunnel in un'infrastruttura complessa.
- La tecnica 4rd (15) prevede l'impiego di tunnel IPv4-over-IPv6 costruiti per associazione stateless di indirizzi IPv4 o coppie indirizzi IPv4/porta sugli indirizzi IPv6 utilizzati come end-point del tunnel. Questo permette ai pacchetti IPv4 di attraversare un dominio 4rd attraverso dei tunnel automatici IPv4-over-IPv6.
- Il Dual-Stack Lite (DS Lite) (16) è una soluzione che unisce l'incapsulamento del traffico IPv4 in tunnel IPv6 con la tecnica Network Address Translation (NAT) e consente a un fornitore di servizi di condividere i medesimi indirizzi IPv4 tra più clienti. DS Lite è un modello dedicato agli ISP che vogliono realizzare un'infrastruttura completamente basata su IPv6 e, nello stesso tempo, vogliono fornire ai loro utenti anche connettività IPv4.
- Infine, 4over6 (17) è un altro meccanismo pensato per consentire a un backbone IPv6 di gestire flussi IPv4. Sotto 4over6 ogni nuvola IPv4 è dotata di un router dual stack che gestisce l'incapsulamento di flussi IPv4, mentre l'estensione Multiprotocol Extension del protocollo Border Gateway Protocol (BGP-MP) (18) (19) è utilizzata per inoltrare i dati necessari alla creazione dei tunnel 4over6 tra i router.

Le soluzioni sopra descritte presentano delle limitazioni che possono influenzare sensibilmente la loro applicabilità in scenari reali. 4in6 è statico e manca di flessibilità nelle infrastrutture di grandi dimensioni in cui l'informazione sugli end-point disponibili è non controllabile in modo centralizzato e fortemente variabile. DS Lite si basa su NAT e presenta il problema del mascheramento dell'indirizzo IP degli utenti finali. In 4rd e 4over6 l'incapsulamento è delegato a nodi specifici, facilitando la configurazione automatica e la gestione dei tunnel, ma introducendo dei punti critici nella rete. Inoltre, 4rd e 4over6 non sono progettati per fornire tunnel tra differenti AS e questo rende impraticabile il collegamento di nodi IPv4 appartenenti a differenti AS se non collegati tramite IPv6.

2.2 La proposta IPv4-in-IPv6

L'approccio IPv4-in-IPv6 proposto nel presente lavoro è basato su quattro aspetti principali: la gestione degli indirizzi IP, l'incapsulamento dei flussi IPv4, la configurazione e gestione dei tunnel e la generazione dei messaggi di errore.

Gestione degli indirizzi IP

IPv4-in-IPv6 richiede l'impiego dedicato di una classe IPv6 di indirizzi pubblici. Nel seguito assumeremo che tale classe sia la 2005::/16. L'utilizzo di tale classe consente la configurazione di tunnel con end-point situati anche su due AS distinti. La generazione di un indirizzo IPv6 avviene prendendo il prefisso prescelto e aggiungendo nella parte meno significativa l'indirizzo IPv4.

Incapsulamento / Decapsulamento

IPv4-in-IPv6 assume che ogni rete di accesso IPv4 disponga di un dispositivo IPv4-in-IPv6, gestito dal provider di servizi Internet (ISP) e che colleghi tale rete al backbone IPv6 (1). Questo dispositivo è responsabile del tunneling IPv4. Quando un pacchetto IPv4 viene inviato attraverso questo dispositivo, quest'ultimo si occupa di incapsulare automaticamente il pacchetto in un pacchetto IPv6. L'incapsulamento è basato su (13), con indirizzi di origine e di destinazione costruiti tramite la regola spiegata al punto precedente. Dopo l'incapsulamento, il pacchetto viene quindi trasmesso attraverso il backbone IPv6. Quando un pacchetto IPv6 raggiunge un dispositivo IPv4-in-IPv6 nella rete di destinazione, questo viene controllato per determinare se contiene un flusso IPv4. Se gli indirizzi sorgente/destinazione del pacchetto appartengono alla classe in uso per flussi incapsulati e se il pacchetto contiene effettivamente un pacchetto IPv4, l'apparato decapsula il pacchetto. Dopo la fase di decapsulamento, il pacchetto viene inoltrato attraverso la rete locale IPv4.

Configurazione e gestione dei tunnel

La configurazione e la gestione dei tunnel è automatica e dinamica, cioè l'instaurazione di un tunnel non richiede protocolli di segnalazione tra i due end-point e non sono richiesti protocolli di controllo per la gestione dei tunnel in caso di riconfigurazione della rete. Ogni tunnel IPv4-in-

IPv6 è basato interamente sui normali meccanismi di routing. La raggiungibilità di una sottorete IPv4 all'interno del backbone IPv6 è consentito dall'annuncio di una determinata rotta (tramite i comuni protocolli di routing) dal relativo router PE che gestisce tale rete IPv4. E' compito del router PE connesso ad una sottorete IPv4 annunciare la raggiungibilità della sottorete IPv6 gestita tramite IPv4-in-IPv6. Per esempio, per permettere il tunneling automatico dei pacchetti IPv4 verso la sottorete IPv4 a.b.c.0/24, il router PE annuncerà se stesso come destinatario dei pacchetti della sottorete 2005::a.b.c.0/120.

Gestione errori mediante ICMP e gestione MTU

Gli errori rilevati fuori dai tunnel IPv6 sono segnalati direttamente con il protocollo ICMPv4 (20). Di contro, gli errori rilevati durante l'invio di un pacchetto IPv6 contenente un pacchetto IPv4 vengono segnalati tramite il protocollo ICMPv6 (21). In questo caso è necessaria la traduzione in ICMPv4 se l'errore si riferisce alla comunicazione del tunnel IPv4. In questo caso, la traduzione viene effettuata dall'end-point sorgente. Inoltre, quando un pacchetto IPv4 entra in un tunnel e la dimensione del pacchetto originale supera la MTU gestita dal tunnel, l'end-point sorgente scarta il pacchetto e restituisce un messaggio ICMPv4 di host non raggiungibile causato da un pacchetto troppo grande. In questo caso il nodo IPv4 sorgente, informato che i suoi pacchetti superano la MTU del tunnel, può reagire riducendo la dimensione dei pacchetti inviati.

2.3 Implementazione

La soluzione proposta è stata implementata sotto forma di un modulo del kernel Linux, e appare come un'interfaccia di rete virtuale. Tale modulo può essere installato all'interno di ogni nodo di rete che vuole supportare direttamente entrambi i protocolli. Nella nostra implementazione, le tabelle di routing dei nodi di incapsulamento rispettano le seguenti regole:

- Nei nodi di incapsulamento IPv4-in-IPv6 la tabella di routing IPv4 è configurata in modo che la rotta predefinita del traffico IPv4 inoltri il traffico verso il dispositivo di rete virtuale. Non appena un pacchetto IPv4 indirizzato a un host remoto raggiunge questo nodo, viene inviato sull'interfaccia virtuale tramite la rotta predefinita. Il modulo kernel IPv4-in-IPv6 entra in azione costruendo il corrispondente pacchetto IPv4-in-IPv6 e lo accoda sullo stack di rete dell'IPv6. Alla fine il pacchetto IPv4-in-IPv6 è gestito dalla tabella di routing IPv6 e trasmesso al prossimo hop all'interno del backbone IPv6.
- Nei nodi di decapsulamento IPv4-in-IPv6 la tabella di routing IPv6 è configurata in modo tale che per ogni rete IPv6 associata ad una rete IPv4 sia presente un'interfaccia di rete virtuale che si occupa di incapsulare e decapsulare i pacchetti di rete. Non appena un pacchetto IPv6 indirizzato a una di tali reti raggiunge il nodo IPv4-in-IPv6, esso lo inoltra al dispositivo di rete virtuale. Il modulo kernel IPv4-in-IPv6 entra nuovamente in azione estraendo il pacchetto IPv4 dal pacchetto IPv4-in-IPv6 e accodandolo sullo stack di rete IPv4. Alla fine il pacchetto IPv4 viene inoltrato al destinatario tramite la tabella di routing IPv4.

3. Prestazioni del modulo sviluppato

Il modulo software è stato sottoposto a sperimentazione per verificarne la correttezza funzionale e le prestazioni in un ambiente di rete reale. In particolare, è stato implementato un test-bed costituito da oltre 50 tra apparati di rete e host di comunicazione. Nel test-bed ogni apparato di rete e ogni host è una macchina virtuale configurata e gestita in un server VMware ESXi. In particolare il test-bed, rappresentato in Fig. 1, è costituito da 4 AS, ognuno implementato in un diverso server VMware. Le connessioni interne agli AS sono virtuali e gestite mediante VMware, mentre le connessioni tra AS sono effettive. La sperimentazione ha riguardato due aspetti fondamentali: la stabilità del modulo software e le prestazioni ottenibili su tunnel IPv4-in-IPv6.

3.1 Stabilità del modulo software

Il primo test effettuato ha riguardato la stabilità del modulo software. Il test ha previsto la configurazione di 3 comunicazioni TCP tra AS differenti (AS2_DC1_HOST1 -> AS1_DC2_HOST2, AS3_DC3_HOST1 -> AS1_DC1_HOST4, AS4_DC1_HOST3 -> AS1_DC1_HOST1) tenute attive per 2 giorni di fila. A valle dell'esperimento è stata effettuata un'analisi dei log di sistema e una verifica dello stato di funzionamento delle macchine coinvolte e del modulo software. Dall'analisi dei sistemi e dei log si è verificato il corretto funzionamento dei sistemi e una comunicazione senza

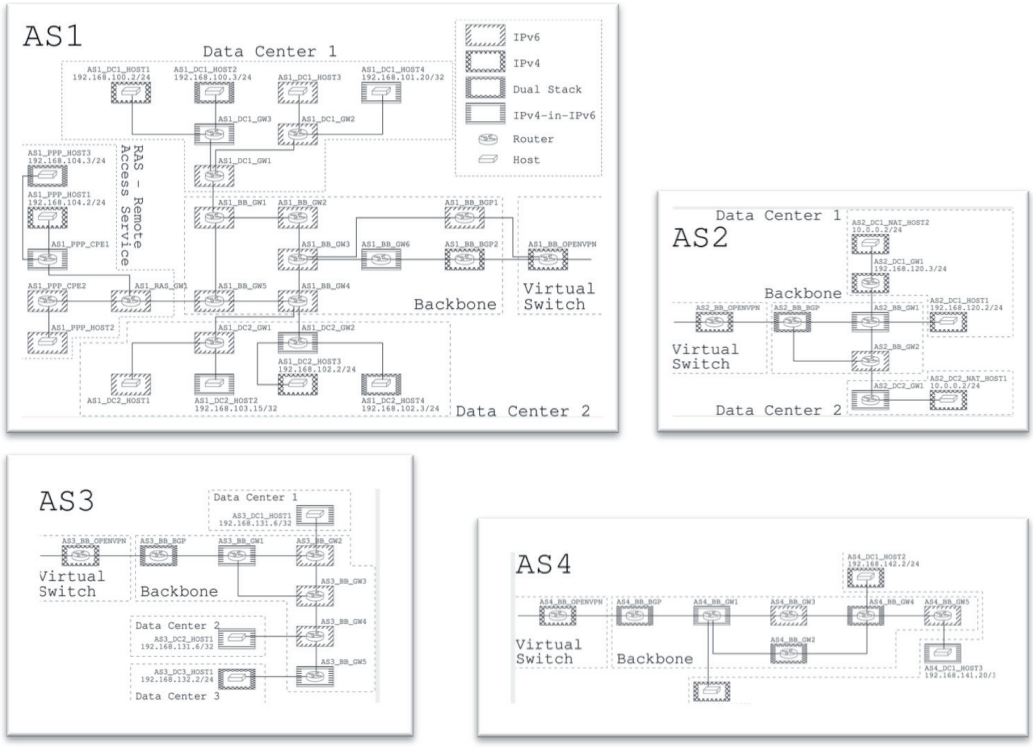


Fig. 1: Test-bed per la valutazione del modulo software

errori al massimo throughput di comunicazione.

3.2 Prestazioni

I successivi test hanno riguardato la stabilità e l'analisi delle prestazioni del modulo implementato. Un primo test è stato realizzato tra due host reali collegati mediante uno switch a 100Mbps. I due host sono stati configurati per comunicare in due condizioni diverse: (i) mediante IPv4 direttamente e (ii) mendiante IPv4 ma passando per il modulo IPv4-in-IPv6. La comunicazione tra gli host ha previsto la creazione di un canale TCP mandato in saturazione mediante un trasferimento FTP. L'analisi del traffico ha evidenziato un calo di prestazioni, imputabile essenzialmente all'overhead di IPv4-in-IPv6, sotto il valore di 1.5% del throughput.

Un secondo test ha riguardato l'instaurazione di diverse comunicazioni tra host dell'AS1 di Fig. 1 al fine di verificare il corretto funzionamento e le prestazioni del modulo software per tutte le possibili combinazioni di incapsulamento. In questo contesto, alcuni host sono stati configurati come server e sono stati dotati di server HTTP (lightHTTP nello specifico), mentre gli host client sono stati configurati per interrogarli e generare un trasferimento dati (mediante wget). I trasferimenti dati sono stati monitorati con tcpdump e registrati. I risultati medi su tre trasferimenti sono riportati in Tab. 1, dove sono anche indicati gli host sorgente

N.	Client	HTTP Server	Throughput (Mbps)-Packet Rate (pkts)	Connessione
1	AS1_DC2_HOST2	AS1_DC1_HOST4	1.74-9928/1.80-10254	IPv4/IPv6
2	AS1_DC2_HOST2	AS1_DC1_HOST2	1.86-10533/1.75-10015	IPv4/IPv6
3	AS1_DC2_HOST2	AS1_DC1_HOST1	1.88-10669	IPv4
4	AS1_DC2_HOST3	AS1_DC1_HOST4	1.73-9864	IPv4
5	AS1_DC2_HOST3	AS1_DC1_HOST2	1.80-10259	IPv4
6	AS1_DC2_HOST3	AS1_DC1_HOST1	1.74-9913	IPv4
7	AS1_DC2_HOST4	AS1_DC1_HOST4	1.71-9801/1.79-10244	IPv4/IPv6
8	AS1_DC2_HOST4	AS1_DC1_HOST2	1.75-9978/1.74-9909	IPv4/IPv6
9	AS1_DC2_HOST4	AS1_DC1_HOST1	1.72-9797	IPv4

Tab. 1 Risultati dei test di valutazione delle prestazioni

e destinazione dei flussi dati. Per tutte le configurazioni, il modulo ha operato in modo stabile, con prestazioni adeguate ai link configurati tra i vari host (10 Mbps).

4. Conclusioni

Nel presente lavoro abbiamo presentato una nuova strategia per la transizione a IPv6, basata sulla costruzione dinamica di tunnel IPv4-in-IPv6. Tale strategia, oltre a essere presentata, è stata implementata e valutata sperimentalmente. Come nota a margine, il modulo software è stato portato con successo in un apparato di rete reale: il FRITZ!Box per WLAN 7390.

Riferimenti bibliografici

[1] K. Nichols, S. Blake, F. Baker, D. Black. "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers" RFC 2474 (Proposed Standard). 1998.

[2] Hinden, S. Deering, R. "Internet Protocol, Version 6 (IPv6) Specification" RFC 2460 (Draft Standard). 1998.

[3] M. Mackay, C. Edwards, M. Dunmore, T. Chown, G. Carvalho. "A scenario-based review of IPv6 transition tools" Internet Computing, IEEE, vol. 7, pp. 27 – 35. 2003.

[4] Carpenter, B. "IPng White Paper on Transition and Other Considerations" RFC 1671 (Informational). 1994.

[5] Haskin, R. Callon, D. "Routing Aspects of IPv6 Transition" RFC 2185 (Informational). 1997.

[6] Gilligan, E. Nordmark, R. "Basic Transition Mechanisms for IPv6 Hosts and Routers" RFC 4213 (Proposed Standard). 2005.

[7] Moore, B. Carpenter, K. "Connection of IPv6 Domains via IPv4 Clouds" RFC 3056 (Proposed Standard). 2001.

[8] A. Durand, P. Fasano, I. Guardini, D. Lento. "IPv6 Tunnel Broker" RFC 3053 (Informational). 2001.

[9] Jung, B. Carpenter, C. "Transmission of IPv6 over IPv4 Domains without Explicit Tunnels" RFC 2529 (Proposed Standard). 1999.

[10] F. Templin, T. Gleeson, D. Thaler. "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)" RFC 5214 (Informational). 2008.

[11] Templin, F. "Transmission of IPv4 Packets over Intra-Site Automatic Tunnel Addressing Protocol (ISATAP) Interfaces" RFC 5579 (Informational). 2010.

[12] Troan, W. Townsley, O. "IPv6 Rapid Deployment on IPv4 Infrastructures (6rd) – Protocol Specification" RFC 5969 (Proposed Standard). 2010.

[13] Deering, A. Conta, S. "Generic Packet Tunneling in IPv6 Specification" RFC 2473 (Proposed Standard). 1998.

[14] Parent, M. Blanchet, F. "IPv6 Tunnel Broker with the Tunnel Setup Protocol (TSP)" RFC 5572 (Experimental). 2010.

[15] R. Depres, S. Matsushima, T. Murakami, O.Troan. "IPv4 Residual Deployment across IPv6-Service networks (4rd) ISP-NAT's made optional". 2011.

[16] A. Durand, R. Droms, J. Woodyatt, Y. Lee. "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion" RFC 6333 (Proposed Standard). 2011.

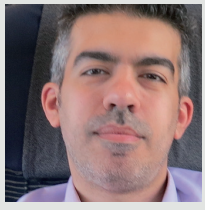
[17] J. Wu, Y. Cui, X. Li, M. Xu, C. Metz. "4over6 Transit Solution Using IP Encapsulation and MP-BGP Extensions" RFC 5747 (Experimental). 2010.

[18] Y. Rekhter, T. Li, S. Hares. "A Border Gateway Protocol 4 (BGP-4)" RFC 4271 (Draft Standard). 2006.

[19] T. Bates, R. Chandra, D. Katz, Y. Rekhter. "Multiprotocol Extensions for BGP-4" RFC 4760 (Draft Standard). 2007.

[20] Postel, J. "Internet Control Message Protocol" RFC 792 (Standard). 1981.

[21] A. Conta, S. Deering, M. Gupta. "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification" RFC 4443 (Draft Standard). 2006



Massimo Vellucci
m.vellucci@unicampus.it

Massimo Vellucci si è laureato al corso di Informatica (Vecchio Ordinamento) presso l'Università La Sapienza di Roma (2006). Attualmente ricopre il ruolo di responsabile delle infrastrutture IT dell'Università Campus Bio-Medico di Roma. L'attività di ricerca si sviluppata principalmente nell'ambito del networking e reti wireless.