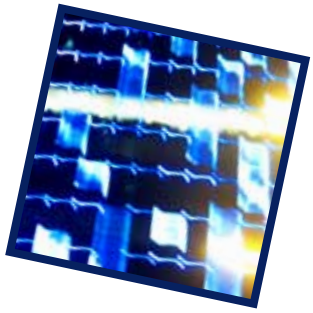
Software-Defined Networking: Esperienze OpenFlow e l'interesse per Cloud



Mauro Campanella[1], Fabio Farina[1], Luca Prete[1], Andrea Biancini[2]

[1]Consortium GARR, [2]INFN – Sezione di Milano Bicocca

Abstract. Il protocollo OpenFlow è divenuto uno standard *de facto* nell'ambito del *Software-Defined Networking* (SDN). L'articolo descrive l'esperienza di ricerca condotta da GARR su tali tecnologie e il potenziale uso di SDN e OpenFlow nelle cloud. È descritta la creazione di un testbed virtuale *OpenFlow* e l'ideazione e realizzazione di un modulo software per l'utilizzo di topologie magliate a livello di data link nelle reti con controller OpenFlow.

1. Introduzione

Con l'avvento del paradigma *cloud* e dell'uso diffuso della virtualizzazione nei sistemi di calcolo, la gestione delle infrastrutture di rete diventa sempre più complessa, richiedendo maggior dinamicità e automazione. La densità e il numero delle risorse fisiche di calcolo disponibili sono in costante aumento grazie agli sviluppi delle microtecnologie. Il paradigma Cloud fa un uso esteso delle tecnologie di virtualizzazione che, slegando le risorse informatiche dalle loro componenti fisiche, ne moltiplica il numero e permette di fruirne in modo più efficace, flessibile e dinamico, per esempio con affidabilità tramite migrazione in tempo reale dei servizi. Lo stesso cambio di paradigma è richiesto e sta avvenendo a livello delle reti che astraggono sempre più dal livello fisico di implementazione e si presentano come infrastrutture virtualizzate.

Per essere efficace, un servizio basato su cloud deve armonizzare la virtualizzazione di calcolo e *storage* con la virtualizzazione delle infrastrutture di rete in ambiente ad alta densità. L'idea di migrare verso reti definite via software viene incontro a questa esigenza. Dal 2008 il nuovo paradigma, chiamato *Software-Defined Networking* (SDN) [1] propone di semplificare i nodi di rete (*router*|*switch*) disaccoppiando il piano d'instradamento (*switching*, realizzato tipicamente in hardware), dal piano di controllo (in software), favorendo l'utilizzo di hardware e software standard e *open source*. Il primo protocollo SDN sviluppato è *OpenFlow*.

SDN permette di programmare a tutti i livelli della pila protocollare le logiche di funzionamento della rete in base ai requisiti di amministrazione, non facendo più unicamente affidamento sui protocolli di uso comune. Un altro vantaggio offerto da tale tecnologia consiste in una gestione semplificata della rete da parte dell'amministratore, grazie a strumenti di *orchestration* centralizzati. Per quanto riguarda i produttori di apparecchiature, l'effetto derivante del modello SDN è una semplificazione delle tecnologie di costruzione di router e switch, con un conseguente abbassamento dei costi.

Le tecnologie SDN come OpenFlow [2][3], attraverso la possibilità di programmare a livello software la rete, introducono la gestione della de-materializzazione delle risorse network e di virtualizzazione. Il modello SDN, quindi, può essere visto come un fattore abilitante per la realizzazione di servizi cloud in cui la rete possa rappresentare un elemento di valore aggiunto e ridurne i costi di realizzazione. Inoltre, la gestione di centri di calcolo e di memorizzazione di massa di tipo cloud è un compito estremamente complesso e articolato. Gli effetti della complessità di gestione sull'infrastruttura di rete sono ampi e SDN può essere un modello per l'otti-

Workshop GARR Calcolo e Storage Distribuito - Selected Papers. Roma 29-30 novembre 2012
ISBN: 978-88-905077-4-8 © 2013 Associazione Consortium GARR

mizzazione dell'uso delle risorse e, più in generale, dell'attività di gestione e amministrazione delle reti nei *data center* cloud.

La principale implementazione del paradigma SDN è OpenFlow. Tale protocollo ha un modello di gestione e funzionamento delle apparecchiature di rete che disaccoppia completamente il piano di *forwarding* da quello di controllo. Oltre a essere stato il primo protocollo SDN disponibile, è aperto e permette di gestire insiemi eterogenei di apparati hardware.

Da circa un anno GARR sta svolgendo un'attività finalizzata alla comprensione e alla valutazione delle potenzialità del protocollo OpenFlow. I primi risultati sono stati la creazione di un testbed virtuale dedicato e lo sviluppo di una variante del protocollo *Spanning Tree* (STP) [4].

Gli aspetti descritti saranno discussi in dettaglio nel seguito dell'articolo, che è organizzato come segue: la prima sezione contiene una descrizione più dettagliata del protocollo OpenFlow; la seconda presenta l'architettura del testbed realizzato; la terza sezione presenta il modulo software sviluppato; l'ultima sezione presenta le conclusioni e i possibili sviluppi futuri.

1. Il protocollo OpenFlow

OpenFlow è un protocollo aperto sviluppato dall'università di Stanford a partire dal 2007, che utilizza il concetto di flusso per la re-direzione dei pacchetti. In questo contesto, per flusso s'intende una sequenza unidirezionale di pacchetti aventi caratteristiche comuni, che attraversa il nodo entro un intervallo temporale, avendo sorgente e destinazione fisse. Attualmente la tecnologia OpenFlow permette di definire un flusso utilizzando caratteristiche dei pacchetti dal livello 2 al livello 4 (compresi) della pila protocollare e supporta Ethernet, IP, TCP e UDP. OpenFlow permette di definire, attraverso l'uso di una maschera, quali siano i campi nell'header del pacchetto in base ai quali esso possa essere considerato parte di uno specifico flusso.

OpenFlow gestisce apparecchiature di *switching* e routing disaccoppiando il piano di controllo da quello di *forwarding*. Infatti, mentre nei router e negli switch standard il piano di forwarding e quello di controllo sono presenti nello stesso apparato, gli switch OpenFlow separano le due funzioni. La parte di *datapath (*chiamata anche di *forwarding* o *switching*) è svolta sugli apparati, mentre quella di controllo è spostata su un controller esterno, tipicamente un server.

Rispetto ai classici dispositivi con CPU e logica integrate, switch, router e access point possono essere così gestiti da un piano di controllo distinto, grazie a uno o più *Network Operating System* (NOS) comuni. Agendo attivamente su questi ultimi si potrà decidere come i flussi siano elaborati e instradati all'interno dell'intera rete.

Il supporto a OpenFlow è già disponibile per diverse apparecchiature di rete, quali switch, router e access point WiFi. Tali dispositivi espongono un'interfaccia standard OpenFlow che permette di interagire con i controller esterni, senza bisogno che i diversi produttori rivelino i dettagli dei loro apparati di rete. OpenFlow è implementato in hardware da una gran parte di essi. Il datapath di un apparato OpenFlow utilizza una tabella di flussi memorizzati; ogni record della tabella contiene una serie di regole che permettono di filtrare determinati pacchetti, quindi i flussi, e applicare a essi un'azione del tipo *send-out-port*, *modify-field* o *drop*. Quando uno switch OpenFlow riceve un pacchetto che non ha mai visto prima, per il quale non vi siano regole di pattern matching attive nella tabella dei flussi, esso manda il pacchetto al *controller,* che decide come gestirlo: può ad esempio scartarlo o aggiungere invece un nuovo record contenente una regola alla tabella dello switch, istruendolo così su quali azioni applicare a pacchetti analoghi in futuro. Secondo le configurazioni scelte sul controller, una regola d'indirizzamento può essere installata in modo proattivo, cioè installata a prescindere dal verificarsi di un evento o in seguito a un evento. Inoltre, ogni regola inserita in tabella può scadere dopo un certo intervallo temporale o essere persistente fino allo spegnimento del nodo.

2. Il testbed virtuale GARR

Da circa un anno si sta svolgendo un'attività finalizzata alla comprensione e alla valutazione delle potenzialità del protocollo OpenFlow. A tale scopo è stato realizzato un testbed virtuale, su un server che utilizza VMware ESXi 5.1 [5], capace di emulare in modo realistico una rete gestita da un sistema OpenFlow.

Nell'infrastruttura sono presenti:

- tre macchine virtuali per la simulazione del traffico utente
- quattro switch virtuali basati su OpenFlow
- due router software per consentire l'accesso a Internet, sia agli *host*, sia agli apparati stessi per manutenzione.

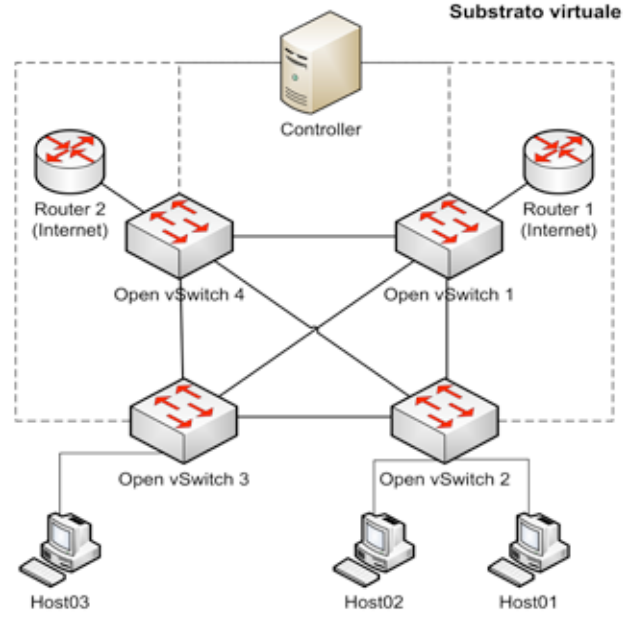


Fig 1 - Schema del testbed virtuale

Il testbed è formato da due piani coesistenti: quello dedicato al traffico utente (rappresentato con linee continue nel diagramma) e quello di monitoring e controllo (linee tratteggiate) dedicato alla comunicazione tra gli apparati OpenFlow e il *controller*.

Si è deciso di focalizzare la ricerca e le prove su protocolli di gestione, per verificare le funzionalità, evitando di tenere conto dei livelli di prestazioni massime raggiungibili, non essendo gli switch in hardware.

Il collegamento tra le porte di rete delle apparecchiature del testbed virtuale è realizzato per mezzo di switch software dell'*hypervisor* con due porte e in "*promiscuous mode*", equivalenti a dei comuni repeater e non partecipanti a OpenFlow.

OpenFlow è implementato da quattro switch emulati da macchine virtuali che eseguono il sistema Open vSwitch e collegati tra loro attraverso una magliatura completa. I possibili cammini tra gli switch sono ridondati per la sperimentazione di meccanismi di *fail-over*, equivalenti di STP, offerti dal protocollo OpenFlow. Ogni switch ha un certo numero di porte, dalle quali transita il traffico utente, collegate ai router, agli host e agli altri switch; inoltre ogni switch OpenFlow ha un'interfaccia di *loopback* per la configurazione e il controllo remoto, la comunicazione con i controller e il collegamento a internet per gli eventuali aggiornamenti.

Nella scelta del controller OpenFlow, si è adottato inizialmente Beacon [6] per le sue grandi potenzialità, affidabilità e diffusione. In seguito, l'attività è stata impostata su FloodLight [7], nato come evoluzione di Beacon e seguito da una comunità utente più ampia e rilasciato sotto licenza Apache2 [8].

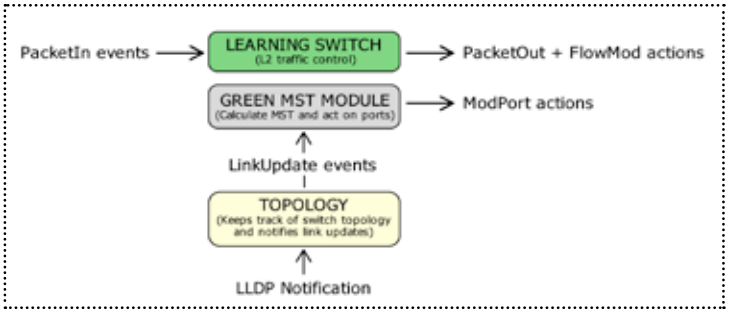


Fig 2 - Il funzionamento dei moduli

3. Sviluppo del modulo GreenMST

Prima di parlare dell'implementa-

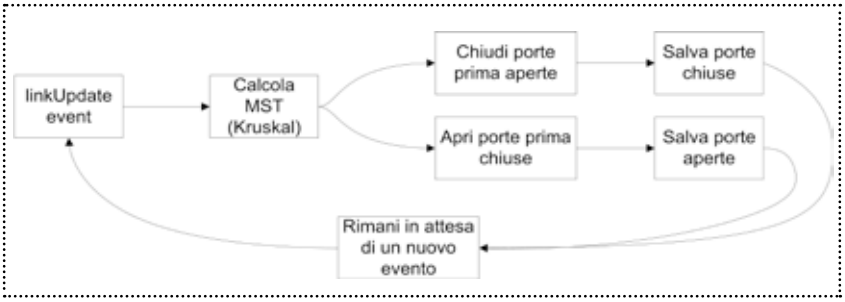


Fig 3 - L'algoritmo Kruskal

zione del modulo *Green Minimum Spanning Tree* (GreenMST) è necessaria una premessa. I controller OpenFlow prevedono la possibilità di usare due moduli diversi per il forwarding dei pacchetti, chiamati *Forwarding* e *LearningSwitch*.
Il primo, più completo e allo stesso tempo più complesso da programmare, prevede l'uso di grafi per astrarre la topologia di rete sul controller centralizzato e supporta l'uso di topologie magliate. Infatti, Forwarding crea un albero di copertura minima per ogni nodo della rete e fa sì che le azioni di *floodling* avvengano solo attraverso le porte appartenenti all'albero, evitando il cosiddetto fenomeno del *broadcast storm*. *Forwarding* è il modulo maggiormente usato e per questo motivo non è generalmente necessario includere nel controller ulteriori tecnologie cosiddette *loop-free*.

Al contrario di Forwarding, LearningSwitch è un modulo semplificato, di facile comprensione, semplice da utilizzare e debuggare e facilmente integrabile con altri moduli, che emula il comportamento dei tradizionali switch di livello 2. Tale modulo, generalmente preferito dagli utenti per la sua praticità, permette di sperimentare una buona parte delle potenzialità offerte da OpenFlow, ma non supporta in modo nativo l'uso di topologie magliate. Quindi, anche se il protocollo Openflow prevede la possibilità di gestire un insieme di switch che utilizzano fra loro il protocollo di *Spanning Tree*, non esiste alcun modulo sui controller in grado di interpretare i messaggi *Spanning Tree*, e i produttori di hardware hanno scelto di non implementarlo, considerando che la maggior parte degli utenti usi il modulo di Forwarding.

Per lo sviluppo del testbed virtuale è stato scelto Learning Switch. Esso mantiene centralmente sul controller le tabelle MAC, tradizionalmente distribuite in locale sugli switch. Analogamente a un comune switch, quando un pacchetto in ingresso arriva al controller, esso memorizza l'associazione tra lo switch di provenienza, la porta d'ingresso e l'indirizzo MAC, poi fa il forwarding del pacchetto al destinatario. Per gestire le topologie magliate è stato sviluppato un nuovo modulo per controller OpenFlow, che fosse in grado di gestire reti con loop a livello 2.

GreenMST è in grado di ricevere le notifiche di alterazione della topologia, dette di *LinkUpdate* (figura a sinistra) emanate dal modulo *Topology* sottostante. A sua volta, Topology intercetta i messaggi di *Link Layer Discovery Protocol* (LLDP) [9] provenienti dagli switch per ricostruire la topologia di rete. A ogni nuovo evento, GreenMST si occupa di calcolare il *minimum spanning tree* del nuovo grafo di rete tramite l'algoritmo di Kruskal [10]. L'albero dei cammini minimi ottenuto, unico per tutta la rete, è usato per chiudere le porte coinvolte nei collegamenti non appartenenti ad esso. Le interfacce escluse sono disattivate attraverso comandi di OpenFlow (PortMod) inviati dal controller agli switch. I cammini attivi disponibili tra tutti i nodi non sono necessariamente ottimali. Tale garanzia si ha solo tra il nodo scelto dall'algoritmo di Kruskal come radice dell'albero e tutti gli altri nodi della rete. L'algoritmo sceglie infatti tra tutti gli alberi possibili quello che minimizza la somma di tutti i cammini per questo albero di copertura minimo.

Per ottimizzare la segnalazione è stata creata un'ulteriore struttura dati che contiene lo stato attuale di tutte le porte degli switch. Al termine del calcolo dello spanning tree, la struttura dati è utilizzata per aprire o chiudere le porte solo quando necessario, minimizzando i messaggi in-

viati serialmente agli apparati.

A ogni link della rete è stato associato un peso statico pari ad uno: in questo modo l'algoritmo privilegia un cammino rispetto a un altro in base al numero di nodi che intercorre tra la radice dell'albero e un nodo di destinazione.

Al momento, il modulo Beacon è disponibile in modalità open source con licenza GPL2 e si sta lavorando per migliorarne i tempi di convergenza e la scalabilità per reti di grandi dimensioni. Presto sarà disponibile anche il modulo per Floodligth con licenza Apache2.

4. Interesse per Cloud

Il paradigma SDN descritto nell'articolo è un elemento abilitante per la realizzazione di soluzioni cloud più mature e articolate. SDN, infatti, introduce due benefici di grande impatto e potenzialmente dirompenti nell'ambito della gestione delle infrastrutture e dei servizi informatici:

1. Da un lato, l'obiettivo specifico di SDN, come dice il nome stesso, è quello di permettere la programmabilità via software di tutti gli apparati di networking. Questa caratteristica permette di sostituire o aggiornare i protocolli standard d'instradamento, adottando quelli più adatti ai centri calcolo.
2. Dall'altro, un controller centralizzato permette di avere un unico punto in cui concentrare la logica di programmazione dell'intera rete, basata sull'analisi per pacchetto dei flussi.

Il primo aspetto permette di superare la classica divisione a strati dei protocolli di rete e ottimizzare l'utilizzo della capacità disponibile, creando nuove architetture di funzionamento dei data center. Tra questi, i data center per le cloud possono beneficiarne, ad esempio ottimizzando il collegamento definito dinamicamente fra i nodi di calcolo e i servizi di storage (ad esempio SAN, iSCSI, AoE).

Grazie alla realizzazione di controller centralizzati è inoltre possibile introdurre un cambio di prospettiva radicale nell'approccio alle applicazioni che fanno un uso intensivo della rete. Ora, infatti, le applicazioni devono adattarsi alle caratteristiche della rete. Con i paradigmi SDN con controller centralizzato come OpenFlow, la rete può essere invece riconfigurata totalmente via software in funzione di quale applicazione genera i flussi di traffico. Questo permette di avere un adattamento programmabile istantaneo che sia in grado di soddisfare le richieste del servizio attivo. Il protocollo OpenFlow si sta evolvendo velocemente, sia al suo interno, sia con compatibilità verso MPLS e altri protocolli. Ai data center il protocollo OpenFlow può offrire una flessibilità di utilizzo delle risorse di rete a circuiti e un'integrazione dinamica con i servizi che non è possibile ottenere con i protocolli precedenti.

Il principale elemento innovativo di SDN risiede nell'aver proposto un paradigma in grado di spostare gli aspetti di configurazione e amministrazione delle reti nel dominio dell'ingegneria software.

Il protocollo OpenFlow e le implementazioni di controller sperimentate hanno comunque ancora limiti rilevanti. Va ad esempio sottolineato che, a oggi, OpenFlow non dispone di estensioni che permettano una comunicazione fra domini diversi. Grazie alle basi poste dal nuovo paradigma SDN è tuttavia possibile immaginare che diverse soluzioni già sperimentate in altri campi informatici possano trovare rapida applicazione anche nel dominio delle reti. Questo è reso possibile dal fatto di aver definitivamente spostato il focus all'interno di architetture software che, come tali, possono essere fatte evolvere in modo più flessibile e rapido.

5. Conclusioni e sviluppi futuri

L'interesse del paradigma SDN e del protocollo OpenFlow consiste nell'apertura delle apparecchiature di rete a un completo controllo dell'utente e alle conseguenti infinite possibilità di composizione di regole e azioni sul traffico. Tali caratteristiche lo rendono particolarmente interessante per i centri di calcolo e i servizi di tipo cloud, che si basano su un'alta densità di apparecchiature e rete trasmissiva, permettendo notevoli ottimizzazioni dell'infrastruttura e del suo

funzionamento.

L'esperienza fatta nel testbed virtuale ha confermato le potenzialità di SDN e del protocollo OpenFlow. Ha permesso di sviluppare in tempi relativamente brevi nuovi algoritmi d'instradamento con funzionalità aggiuntive a quelle classiche. L'innovazione nel campo dei protocolli delle reti locali di tipo Ethernet è un campo attivo della ricerca nelle reti, a cui SDN può dare strumenti nuovi.

Il modulo creato permette di sfruttare le potenzialità offerte da LearningSwitch anche in topologie magliate. Allo stesso tempo, spegnendo le interfacce fisiche degli apparati si può realizzare, in linea di principio, un risparmio energetico per i datacenter che ne faranno uso. Nella progettazione del modulo si è compreso come si possa pensare di usare concretamente parametri aggiuntivi per l'instradamento dei pacchetti. Si potrebbe assegnare dinamicamente un peso a ogni circuito (quindi a ogni arco del grafo) in base alle caratteristiche delle interfacce e della congestione della rete, come carico del circuito e ritardo di trasmissione banda.

Il passo successivo sarà la progettazione di moduli più complessi e l'analisi dell'efficacia di OpenFlow in un dominio di rete di produzione. Si studieranno le capacità di gestione di un gran numero di apparecchiature di livello 2 e livello 3, definendo policy multi-livello, quali ad esempio *routing*, *firewall* e *traffic engineering*.
Il protocollo OpenFlow è in continua evoluzione: ben presto sarà disponibile una nuova versione capace di supportare IPv6 e livello ottico (con controllo delle lunghezze d'onda), oltre al *tagging* dei pacchetti per il supporto a MPLS e QinQ.

Lo sviluppo e la gestione di tale ambiente restano per ora ancora troppo complessi per un uso diffuso. Il familiarizzarsi con l'architettura OpenFlow e la realizzazione dei moduli per i controller sono infatti operazioni onerose che richiedono per ora la collaborazione di personale qualificato ed esperto sia nella programmazione che nel networking.

Riferimenti bibliografici

[1] Nick McKeown, "Software-defined Networking", Infocom Keynote Talk, April 21st 2009, Rio de Janeiro, Brazil - disponibile ad http://tiny-tera.stanford.edu/~nickm/talks.html

[2] OpenFlow white paper: www.OpenFlow.org/documents/OpenFlow-wp-latest.pdf

[3] Wiki OpenFlow: http://www.OpenFlow.org/wk/index.php

[4] http://tools.ietf.org/html/rfc4318

[5] http://www.vmware.com/products/vsphere-hypervisor/overview.html

[6] Beacon controller: https://OpenFlow.stanford.edu/display/Beacon/Home

[7] FloodLight controller: http://floodlight.OpenFlowhub.org

[8] http://www.apache.org/licenses/LICENSE-2.0.html

[9] http://standards.ieee.org/getieee802/download/ 802.1AB-2009.pdf

[10] Sedgewick, R. and Wayne, K, "Algorithms (4th Edition)", Ed. Addison-Wesley Professional, 624-627, 2011

Mauro Campanella
mauro.campanella@garr.it

Laureato in Fisica a Milano nel 1985. Svolge l'attività per la rete della ricerca italiana (Consortium GARR) in cui ricopre il ruolo di coordinatore dell'attività di ricerca e sviluppo.
Ha sempre collaborato a progetti internazionali. Oltre alla partecipazione alla creazione delle varie generazione della dorsale europea della ricerca (GÉANT) ed ai suoi servizi (premiumIP, AutoBAHN, Perfsonar) è stato coordinatore del progetto FEDERICA di supporto a Future Internet.