GARR

The Italian Academic & Research Network

www.garr.it

Software Defined Networking

Esperienze OpenFLow e l'interesse per i servizi Cloud

Luca Prete, Mauro Campanella

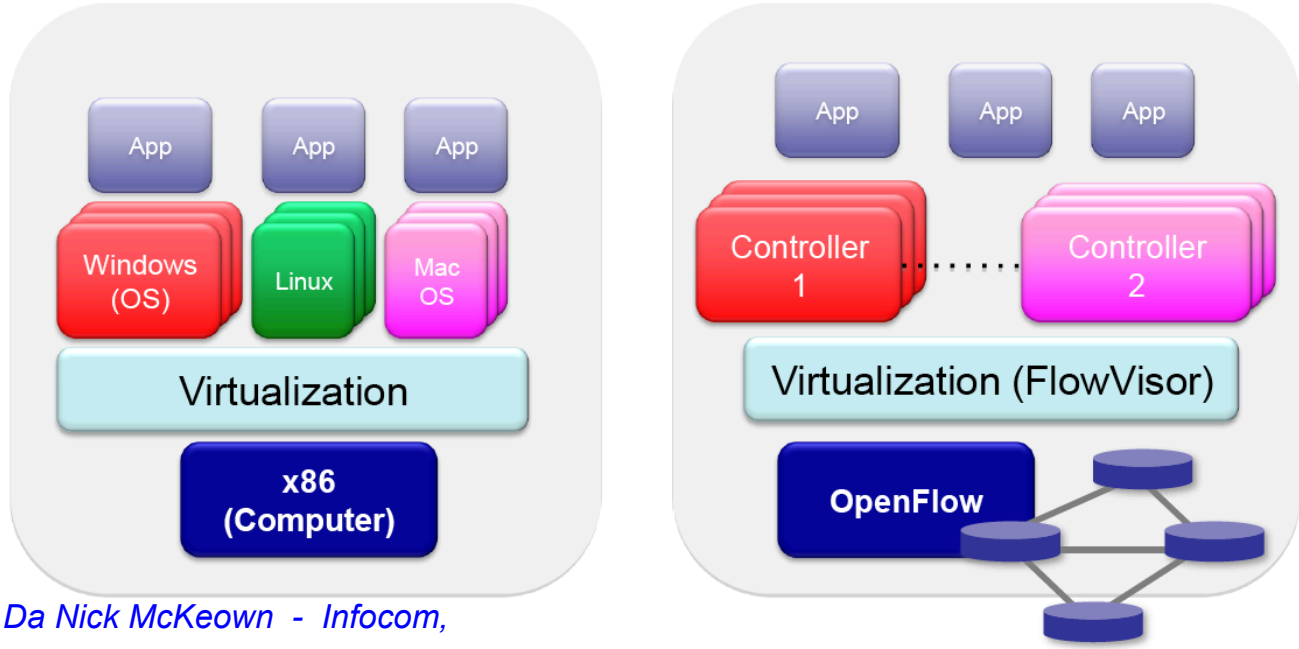
Workshop GARR - Calcolo e storage distribuito, Roma, 30.11.2012

Consortium GARR

AGENDA

- Software Defined Network e clouds
- OpenFlow, esperienza GARR
- Presentazione dell' attività svolta
  - Struttura del test-bed virtuale
  - Funzionalità e casi d' uso implementati

Software Defined Networking



Da Nick McKeown - Infocom, Brasile, Aprile 2009

La proposta originale è di semplificare i nodi di rete (router/switch) :

- disaccoppiando la parte hardware (switching) dal software (controllo)
- Usando hardware standard e SW OpenSource

E' così possibile accellerare l'innovazione all'intero della rete (HW e SW).

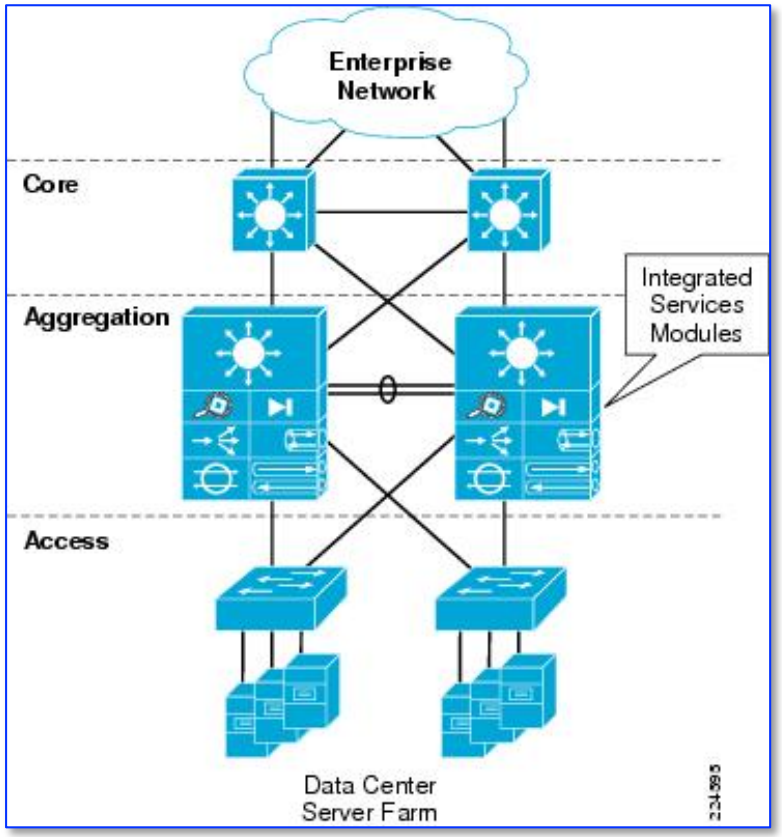
Un' implementazione di SDN - OpenFlow

- OpenFlow è proposto da Stanford nel 2007 come esempio di SDN
- Ad oggi è considerato dalla maggior parte dei produttori hardware e software uno standard de facto, un concorrente *per un singolo dominio* a MPLS

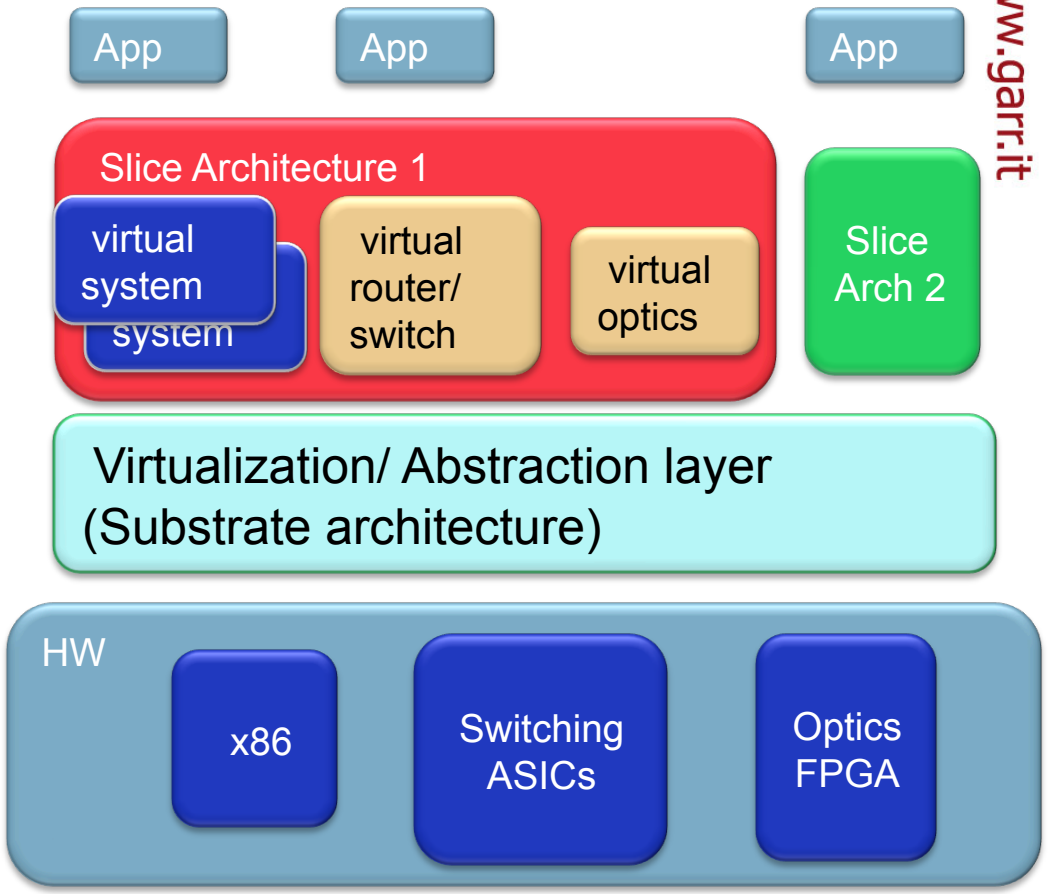
- Trattamento del traffico basato sul concetto di "flusso"
  - Un flusso è una sequenza di pacchetti identificabili da uno od un insieme di 'etichette' comuni (indirizzo IP, MAC address, porta,…)
  - Il nodo di rete fa solo forwarding

- Piano di controllo (esterno ai router/switch switch) disaccoppiato da quello di forwarding.
  - Un computer esterno intelligente + tanti switch/router "semplici"

Cloud e Software Defined Networking



Networking tradizionale in un data centre

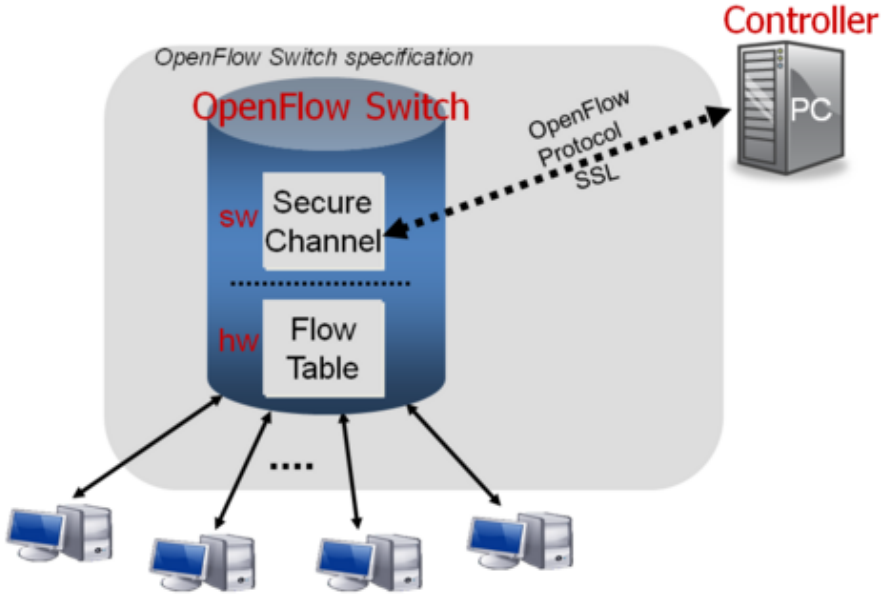


I principi base del networking (affiidabilità, capacità, ridondanza, sicurezza,..) sono contemporaneamente semplificati ed estesi.

AGENDA

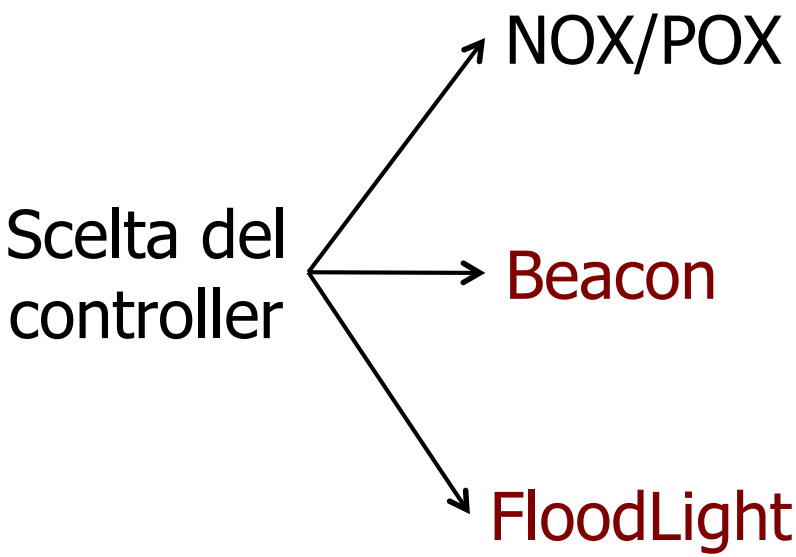
- Software Defined Networks e clouds
- OpenFlow, esperienza GARR
- Presentazione dell' attività svolta
  - Struttura del test-bed virtuale
  - Funzionalità e casi d' uso implementati

Un' implementazione di SDN - OpenFlow



- Interfaccia standard verso i device di diversi produttori (HW e SW)
  - Tanti switch scambiano messaggi OpenFlow standard con un "controller" centralizzato attraverso un canale sicuro (TCP/SSL)
- Tabelle dei flussi: regole + azioni + statistiche
- Se un pacchetto non fa parte di un flusso noto, è inviato al controller che decide cosa fare
- I flussi scadono

I controller OpenFlow



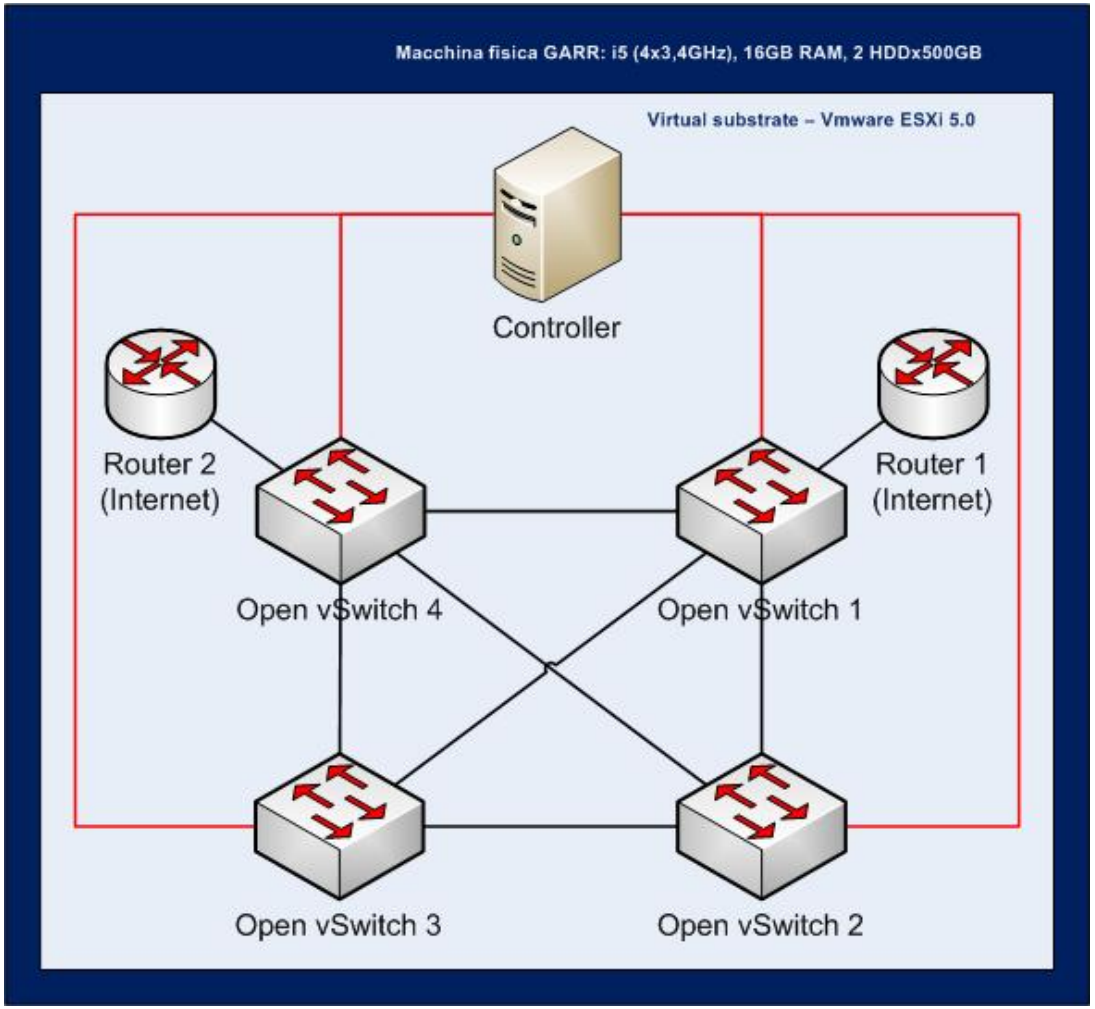
- Perché?
  - Java
    - Multi-piattaforma
    - Thread-oriented
  - Usati in test-bed di grandi dimensioni (+150 switch)
  - Modularità

Controller: Instradamento del traffico

- La logica con la quale sono instradati i pacchetti nella rete è implementata centralmente nel controller e quindi le regole inviate ai nodi rete

- I controller prevedono due metodi per l'inoltro dei pacchetti:
  - LearningSwitch: Emulazione di un switch L2
    - Le tabelle dei MAC sono memorizzate nel controller
    - Problema: non si possono gestire i loop nativamente (se non supportato dagli switch)
  - Routing:
    - Non è il routing che conosciamo, è la vera novità introdotta da OF!
    - Pacchetti instradati (a flussi) grazie ad una conoscenza globale della rete
    - Mappa è mantenuta aggiornata grazie a Device e Topology manager
    - Topology si occupa dei collegamenti switch – switch
    - Device memorizza i punti di connessione device – switch

Il test-bed virtuale



- Obiettivo: sperimentazione delle potenzialità offerte da OpenFlow

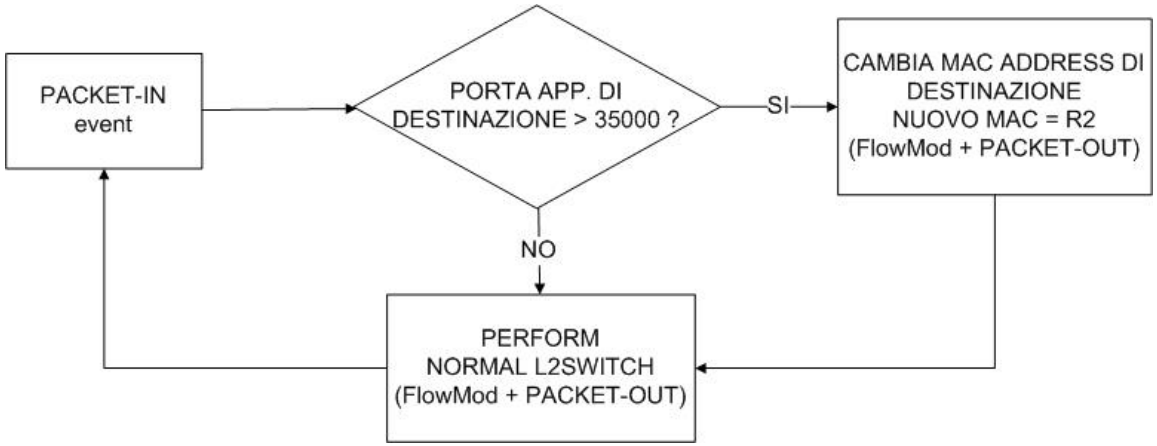
Use Case 1 - Instradamento del traffico in base all'applicazione

- Problema:
  - Fino ad ora costoso/complesso re-direzionare i pacchetti verso gateway diversi in base al loro contenuto.

- Obiettivo use case 1:
  - Re-direzionare il traffico di rete verso gateway differenti in base all'analisi del traffico
  - Se possibile, i pacchetti devono sempre seguire il percorso minimo per arrivare al router di destinazione

- Soluzione:
  - Discriminare il tipo di traffico attraverso un'analisi multi-layer
  - Filtrare i pacchetti in ingresso al controller prima che siano processati da Learning Switch e cambiare il MAC address di destinazione.

Test: traffico di “ricerca”- OK



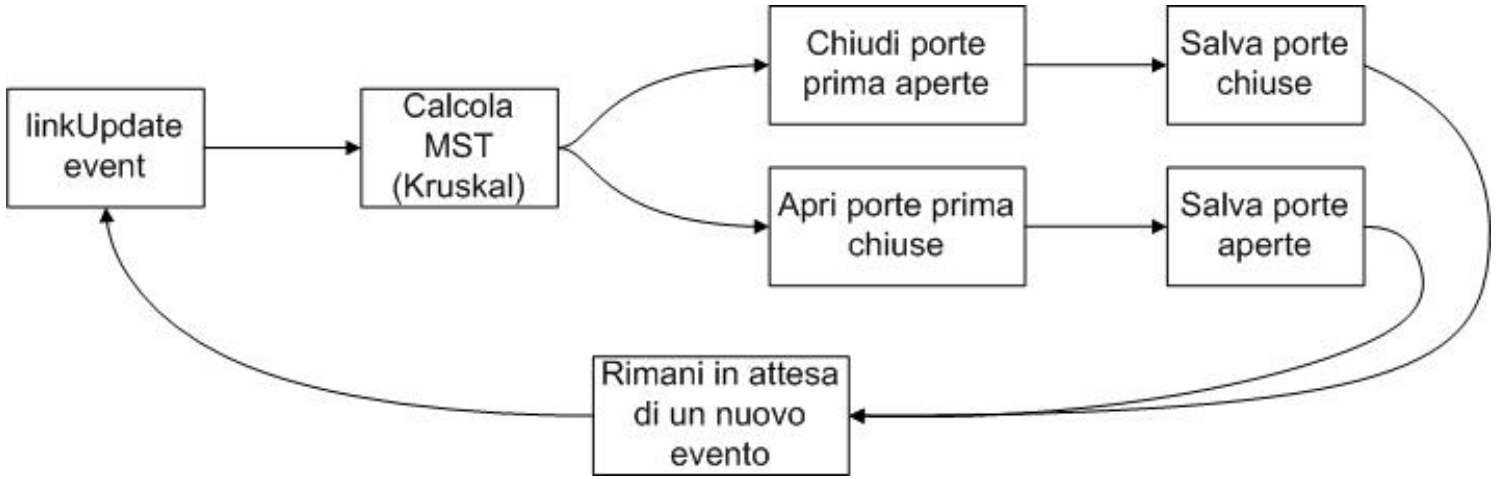
- Due router
  - Router primario per best-effort
  - Router secondario per traffico della ricerca

- Pacchetti diretti normalmente verso il router primario (def. gateway)
- Se porta TCP_dst superiore a 35000 (tipico grid)
  - Cambio indirizzo MAC address di destinazione e instrado il pacchetto con L2Switch

- Niente porte di uscita statiche
  - Funzionalità di L2 switch rimangono! (MAC table)

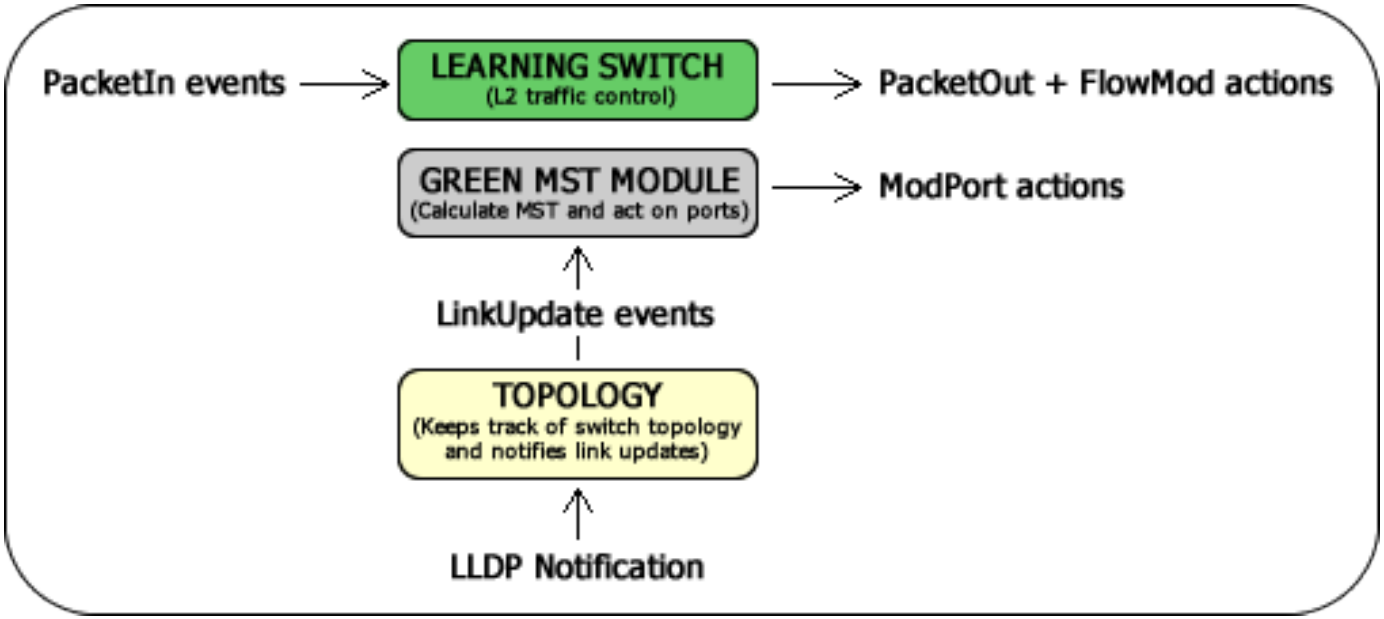
Use case 2 - Loop-free e fast-failover - OK

- Problema: Il modulo di L2Switch dei controller non supporta i loop di rete (a meno che non lo facciano nativamente gli switch)
- Obiettivi:
  - Usare L2Switch con looped topologies
  - Offrire funzionalità di recovery e fail-over
  - Additional requirement: risparmiare energia
- Soluzione:
  - Sfruttare il modulo topology per calcolare il Minimum Spanning Tree della rete e spegnere le porte che non ci servono



Implementazione in Beacon

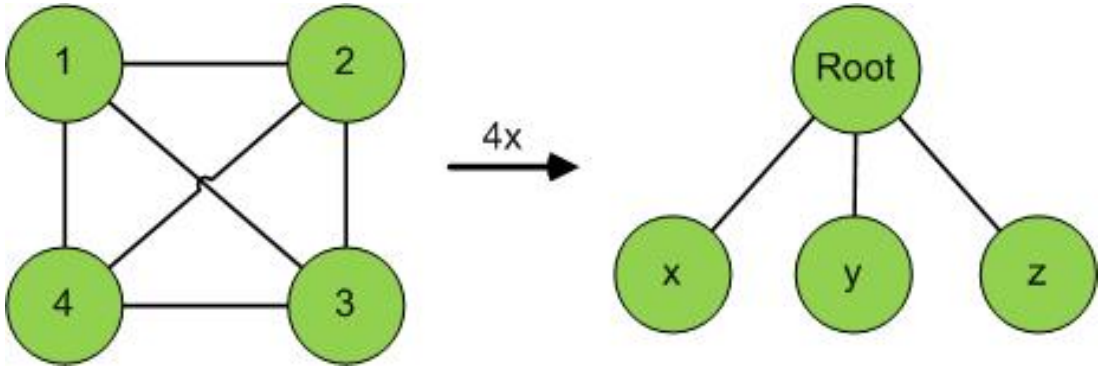
- Creazione di un nuovo modulo: GreenMST
  - Input: eventi di cambio topologia tra gli switch (Topology)
  - Costruzione di MST (Kruskal)
  - Modifica dello stato delle porte attive attraverso comandi OF
  - Minimizzazione comandi inviati agli switch (ricordo lo stato precedente)



- Gli L2 switch fanno il forwarding dei pacchetti solo attraverso i percorsi minimi, grazie alle porte aperte a loro disposizione

Use case 3 - Floodlight e modulo di routing - OK

- Modulo di routing: evololuzione del modulo di LearningSwitch
- L'algoritmo di default nel controller:



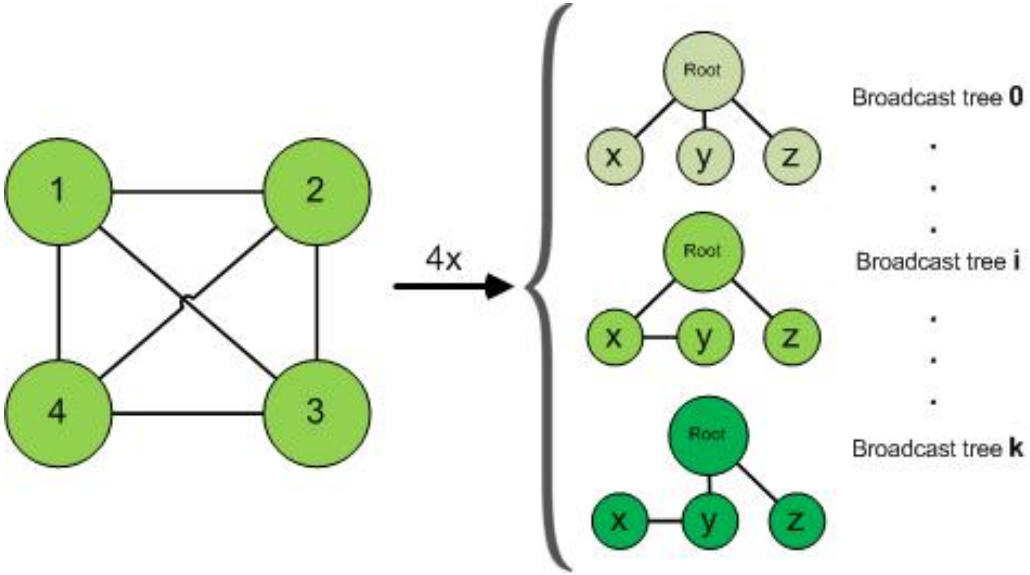
- Per ogni nodo della rete costruisce l'albero di broadcast
- Installa adeguatamente i flussi sugli switch
- Niente loop, flooding solo su albero di broadcast

- Benefici:
  - Superato il concetto di Spanning Tree Protocol
  - Ho sempre la mappa aggiornata della rete grazie ai moduli Device e Topology
- Problema:
  - In caso di recovery devo ricalcolare tutti gli alberi di broadcast

Soluzione: K-Shortest path algorithm

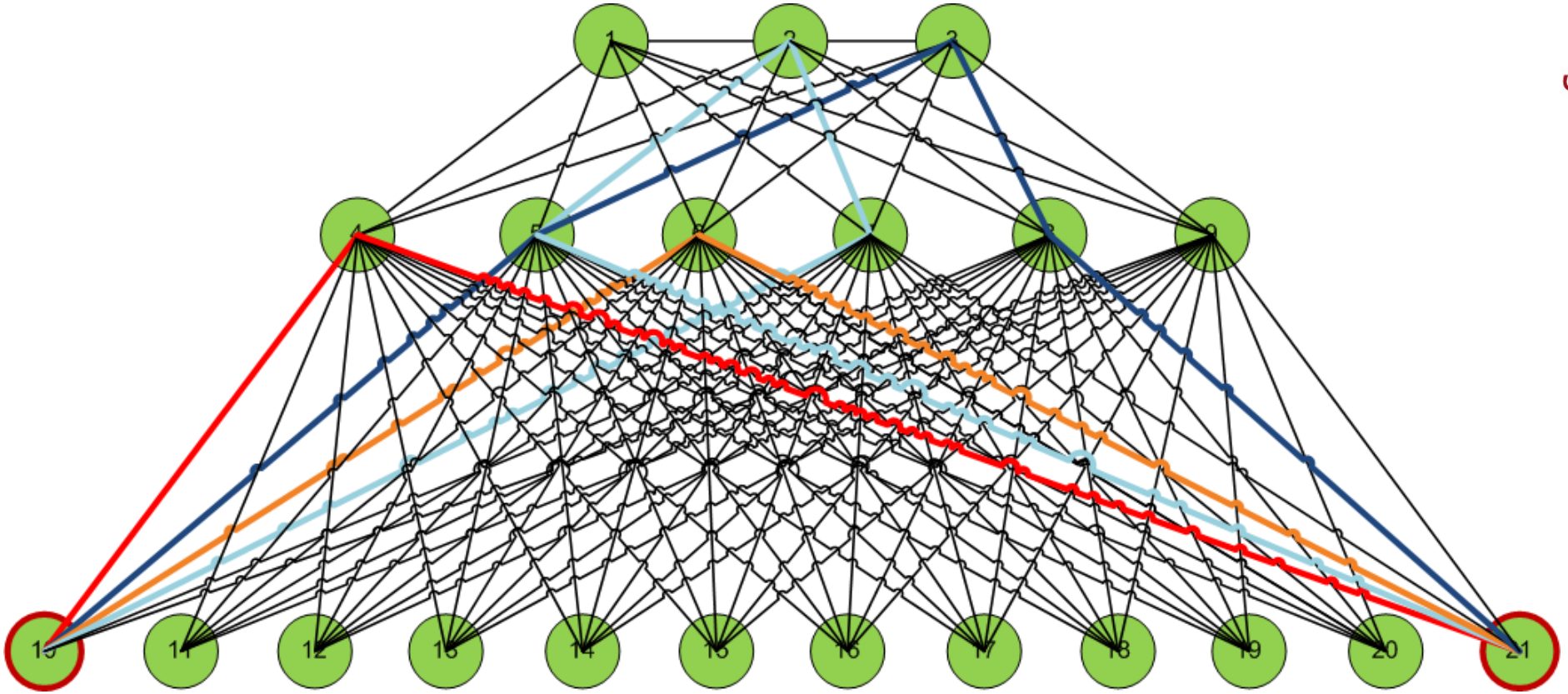
- Obiettivo:
  - Non dover ricalcolare ad ogni cambio di topologia gli alberi di broadcast per ogni nodo di rete
- Soluzione:
  - Adozione dell' algoritmo di K-Shortest-Path (KSP): per ogni coppia di nodi calcola i k cammini minimi in ordine di peso

- Nuovo algoritmo:



- Cache
- Costruzione dinamica broad. tree
- Calcola KSP per ogni nuovo nodo
- Se un percorso si guasta annullalo
- Se tutti i k percorsi sono invalidati per una coppia di nodi ricalcola KSP

Un caso d'uso realistico



- L'algoritmo ha calcolato i primi quattro percorsi tra il nodo 11 e il nodo 21
- In caso di fallimento del percorso minimo (rosso = 2 hop)
  - Selezionato il percorso minimo disponibile (arancione = 2 hop)
  - …e così via

Conclusioni

- In tutti i casi di uso OF ha dimostrato la sua flessibilità e capacità di innovazione. Lo sviluppo del test-bed e dei protocolli ha permesso in tempi brevi di acquisire la conoscenza del protocollo OF e di contribuire al suo sviluppo (Feedback positivi da partecipanti conferenza EWSDN, anche commerciali).

- Validato ampio spettro applicativo:
  - È possibile re-direzionare i flussi di traffico in base all' analisi dei loro pacchetti
  - Si possono usare tecnologie loop-free con LearningSwitch, anche con switch OpenFlow non muniti di STP
  - Grazie al modulo GreenMST si può risparmiare energia all' interno dei DC
  - Possibile ottimizzare migliorate le tempistiche e le performance, in caso di recovery, del modulo di routing

- Il protocollo permette infinite possibilità grazie alla introduzione dello strato SW (SDN) che va adeguatamente progettato e sviluppato.

Prossimi passi

- Valutazione di un test-bed con risorse anche fisiche
  - Partecipazione a progetti europei
  - Sviluppo di eventuali collaborazioni
- Introduzione di FlowVisor: Ambiente multi-controller distribuiti
  - Sperimentazione
  - Analisi della sicurezza, isolamento del traffico
- Semplificazione dell'esperienza utente
  - Interfaccia web + client desktop
  - Analisi dei principali tool di amministrazione e debug
- Integrazione dei dati provenienti dal monitoring passivo
  - SNMP
  - Database + Netflow

GRAZIE

Domande

Referenze

- Software Defined Networking
  - Nick McKeown, "Software Defined Networking", Infocom April 2009, Brasil
  - Open Networking Foundation (ONF):

- Sito originale OpenFlow: http://www.openflow.org

  Controllers:
  - Beacon: https://openflow.stanford.edu/display/Beacon/Home
  - Floodlight: http://floodlight.openflowhub.org
- Open vSwitch: http://www.openvswitch.org

- EWSDN: www.ewsdn.org

Il lavoro di ricerca svolto su OF

- Borsista GARR 2011/2012
  - Scopo della borsa: studiare le potenzialità delle SDN, in particolare del protocollo OpenFlow

- Fasi di lavoro
  - Studio del protocollo e degli strumenti tecnologici
  - Sviluppo di un test-bed di rete virtuale installato localmente su hardware di GARR ospitato a INFN - Milano Bicocca
  - Sviluppo di tre casi d'uso
  - Raccolta dati e analisi dei risultati
  - Partecipazione a workshop e stesura articoli (2012)
  - Stesura documentazione e report trimestrali