Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea in Scienze di Internet

OPENFLOW E SOFTWARE-DEFINED NETWORKING: L'EVOLUZIONE DELLA RETE PROGRAMMABILE

Relatore:
Chiar.mo Prof.
MARCO ROCCETTI

Presentata da:
ALESSANDRO FORGIONE

II Sessione
Anno Accademico 2013/2014

INDICE

1. INTRODUZIONE

1.1 La necessità di una nuova rete

Al giorno d'oggi la rete presenta una struttura piuttosto complessa composta da numerosi dispositivi come switch, router o middlebox, che permettono lo smistamento dei pacchetti contenenti i dati scambiati dagli utenti (*host*) connessi tra loro. Dall'altro lato, gli operatori che forniscono il servizio di connessione si trovano a dover fare i conti con la configurazione manuale di protocolli e regole altamente sofisticati per garantire il funzionamento corretto delle numerose applicazioni che fanno uso della rete. Tuttavia, la strumentazione necessaria per effettuare queste operazioni è piuttosto limitata e il più delle volte si riduce ad un intervento fisico sui dispositivi connessi alla rete, cambiandone di conseguenza la sua struttura. Tutto ciò comporta regolarmente diversi fattori di rischio in termini di performance, ma soprattutto di stabilità del sistema. Il verificarsi di un errore critico conduce a sua volta ad una serie di operazioni di manutenzione e risoluzione che, proprio a causa di questa struttura complessa, possono richiedere tempistiche smisurate in relazione al volume di traffico presente nelle reti odierne.

La maggior parte delle reti convenzionali è formata da una configurazione gerarchica, dato che ai tempi dell'esplosione di Internet e del world wide web veniva prediletta la struttura "client-server", cioè un'architettura di rete che permette agli utenti connessi detti anche "client" l'accesso a informazioni e risorse contenute in un altro componente chiamato "server". L'utilizzo odierno delle reti però non calza particolarmente con questo sistema di interazione. L'avvento di elementi come i Data Center, cioè dei centri di elaborazione dati a capienza elevata che permettono a privati e aziende di immagazzinare mediante la rete enormi quantità di dati, ha portato ad un forte cambiamento nella direzione del traffico, in quanto qualsiasi utente connesso alla rete è in grado di trasmettere contenuti di dimensioni elevate attraverso un comune dispositivo come un PC o un tablet, cancellando così questa netta distinzione tra "client" e "server".

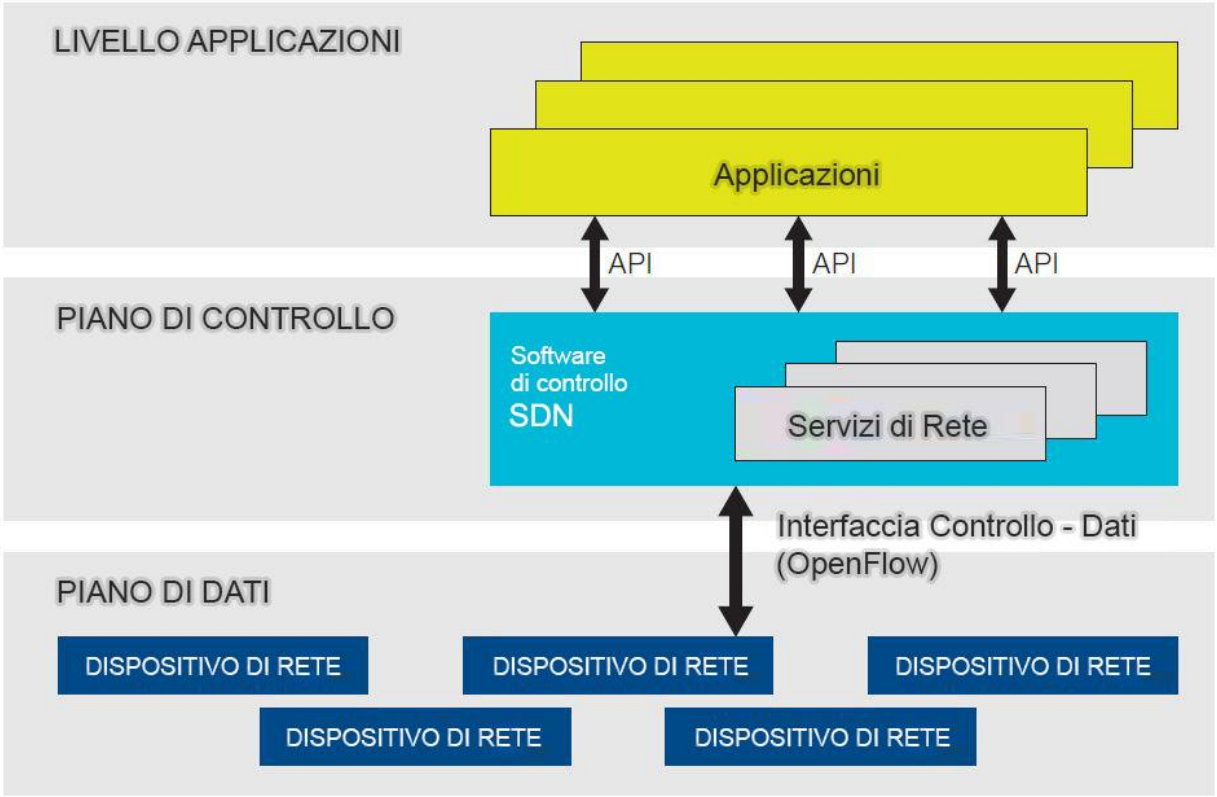
Un caso sempre più presente nella vita di tutti i giorni riguarda senz'altro i servizi cloud, in gergo "Cloud Computing" che, grazie all'utilizzo di dispositivi come i Data Center, consentono agli utenti l'accesso alle proprie risorse e applicazioni da qualunque posizione geografica e in ogni momento. Questo servizio richiede, oltre a un traffico più agile e dinamico di quello offerto dalla gerarchia "client-server", un ambiente ad alta sicurezza e stabilità, ad elevata scalabilità e con politiche più reattive.

Lo sviluppo di nuovi servizi e applicazioni ha ostentato a ricercatori e programmatori una forte rigidità nell'implementazione di questi sulle reti odierne, conosciuta come "ossificazione di Internet" (*Internet ossification*[1]). Per soddisfare i bisogni quotidiani di privati e imprese è necessario lo sviluppo costante di nuovi servizi e protocolli sempre più complessi che garantiscano maggiore affidabilità, connettività e sicurezza. La staticità della rete comporta perciò una forte limitazione di fronte all'ambiente di natura dinamica appena descritto. Di conseguenza, anche la modifica di un elemento appartenente alla rete può comportare la riconfigurazione manuale degli altri dispositivi connessi come ad esempio switch, routers e middlebox.

[1] M. Mendonca: *A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks*, IEEE Communications Surveys & Tutorials, 2013, pp. 1-2

1.2 Introduzione alle reti programmabili, al modello Software-Defined Networking e OpenFlow

L'idea di un'architettura programmabile risponde a tutte le richieste rimaste insoddisfatte a causa dei limiti delle reti tradizionali e può rappresentare la vera soluzione destinata a semplificare l'evoluzione della rete come la conosciamo oggi.



Il modello di una rete Software-Defined Networking (*SDN*) prevede una struttura in cui il piano di controllo (*control plane*) è separato dal piano di dati (*data plane*) e quindi è direttamente programmabile. In altre parole il controllo del traffico, che attualmente risiede in ognuno dei dispositivi di commutazione connessi, viene trapiantato su uno o più calcolatori che, mediante applicazioni specifiche, sono in grado di amministrare dinamicamente la rete come un'entità logica e virtuale.[2] La porzione logica della rete si accentra quindi in un'unica entità dotata di una visuale panoramica di tutta la struttura, rendendo tutti i dispositivi di commutazione come switch o router semplice hardware di inoltro.

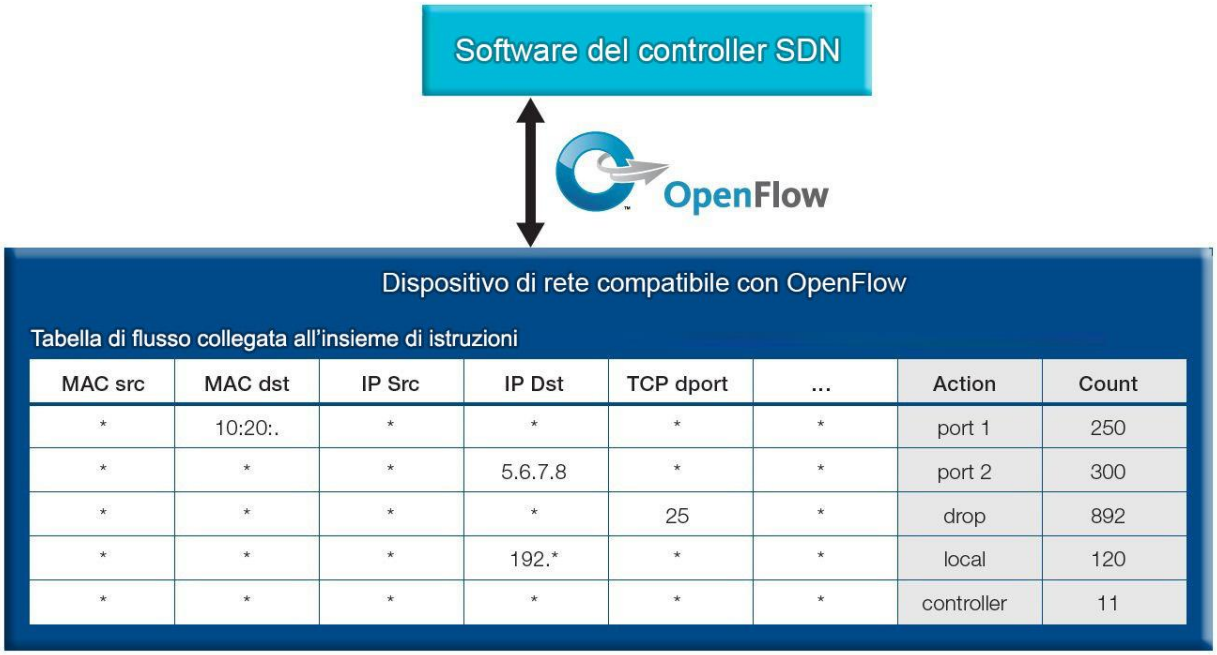
Il paradigma SDN può soddisfare perciò quel bisogno di flessibilità e dinamicità adatte per la gestione, configurazione e sicurezza delle informazioni e risorse condivise dai nuovi servizi e applicazioni che si sono sviluppati nell'ultimo decennio.

I router e gli switch sono tipicamente sistemi "chiusi", ovvero prodotti con interfacce di controllo limitate e sviluppate ad-hoc dall'impresa di provenienza, perciò con licenza proprietaria. Per questa ragione si è resa necessaria la creazione di un'interfaccia di comunicazione "aperta" in grado di instaurare una comunicazione

[2] ONF White Paper: *Software-Defined Networking: The New Norm for Networks*, Open Networking Foundation, 2012, pp.7-8

tra controller e i dispositivi suddetti. Nel 2006 un team di ricercatori della Stanford University ha sviluppato "Ethane"[3], un prototipo di interfaccia che introdusse il concetto di "flusso" (*flow*), diventata in seguito l'unità di misura convenzionale della trasmissione dati nelle reti basate su SDN.

Dopo appena tre anni, sviluppato all'interno dello stesso campus e in seguito adottato dall'organizzazione Open Networking Foundation (*ONF*), nasce il successore naturale di Ethane: OpenFlow, l'attuale interfaccia programmabile standard tra controller e dispositivi di inoltro. OpenFlow permette un accesso diretto ai piani di inoltro di switch e router affinché sia possibile monitorare e gestire il traffico della rete secondo determinate necessità. Il concetto di flusso permette di identificare specifiche porzioni di traffico con delle regole impostate in precedenza e immagazzinate in tabelle denominate tabelle di flusso (*flow table*) in modo da far intraprendere azioni personalizzate. Ciò comporta inoltre la possibilità di controllare la rete a livello atomico, ma soprattutto di ottenere una reattività alle modifiche da parte dell'utente in tempo reale.

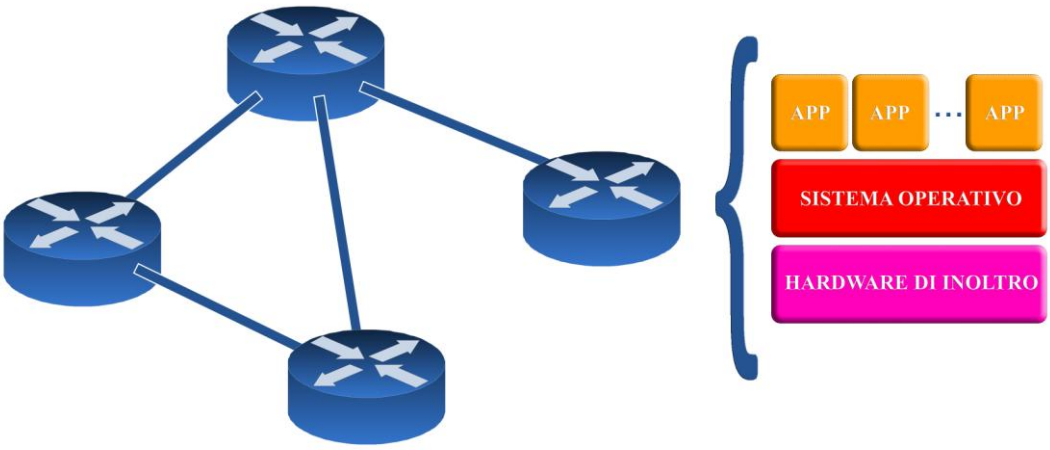


MAC src	MAC dst	IP Src	IP Dst	TCP dport	...	Action	Count
*	10:20:.	*	*	*	*	port 1	250
*	*	*	5.6.7.8	*	*	port 2	300
*	*	*	*	25	*	drop	892
*	*	*	192.*	*	*	local	120
*	*	*	*	*	*	controller	11

3 M. Casado: *Ethane: Taking Control of the Enterprise*, Sigcomm, 2007

1.3 Differenze rispetto alle reti tradizionali

Oltre alle peculiarità evidenti già citate, come l'architettura basata su un piano di controllo centralizzato e scisso dal piano di dati dei dispositivi di commutazione piuttosto che una struttura in cui la gestione del transito di un pacchetto di dati è distribuita da ognuno di questi macchinari, SDN presenta la sostanziale differenza costituita dalla presenza di applicazioni dotate di coscienza globale della rete, e non viceversa. Le reti tradizionali non inviano informazioni e dettagli sullo stato di sé alle applicazioni che ne fanno uso, mentre con un approccio basato su SDN e OpenFlow i programmi non solo sono in grado di monitorarne lo stato, ma hanno il potenziale di adattarne la configurazione secondo le proprie esigenze.



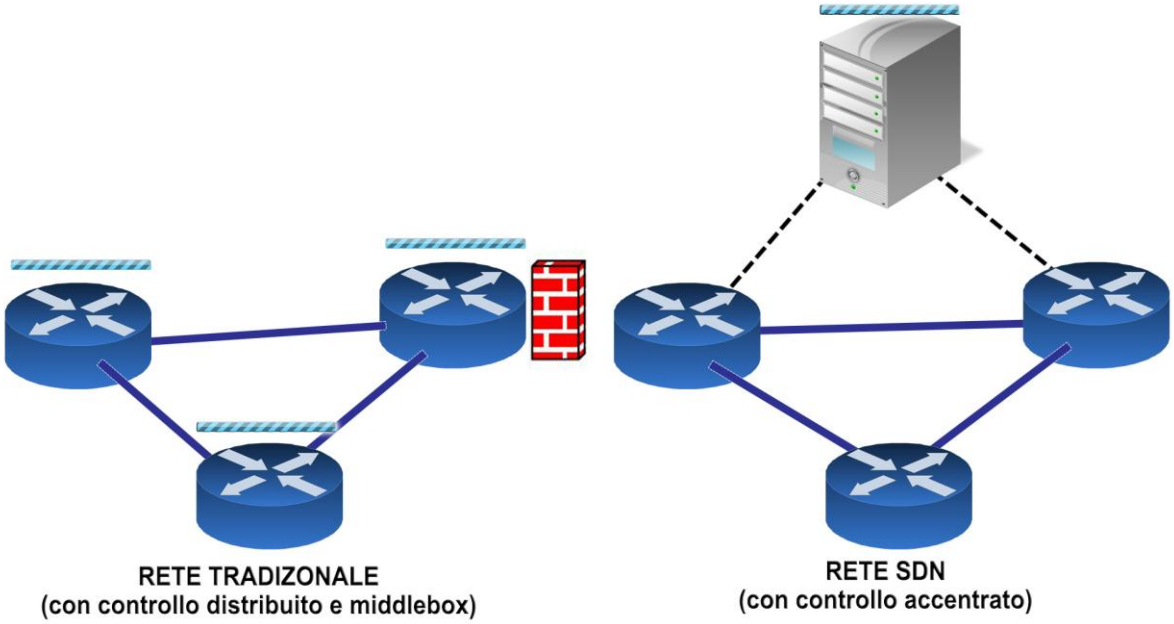
La presenza dell'unità chiamata controller comporta un'altra caratteristica distintiva dell'architettura SDN e rappresenta il cosiddetto piano di controllo logico centralizzato. Tuttavia il controller non è necessariamente accentrato fisicamente: per ragioni dovute a livelli di performance, affidabilità o scalabilità, il controller SDN può essere distribuito in modo da essere composto da diverse unità controller fisiche cooperanti tra loro. Ciononostante, il piano di controllo resta comunque un unico elemento logicamente centralizzato che agisce come un sistema operativo di rete, cioè con mansioni di scheduling, risoluzione di conflitti e corretta coordinazione della strumentazione hardware.[4]

[4] Per un approfondimento si veda: *SDN Architecture Overview*, Open Networking Foundation, 2013, p. 5

2. STRUTTURA DI RETE BASATA SUL PARADIGMA SDN

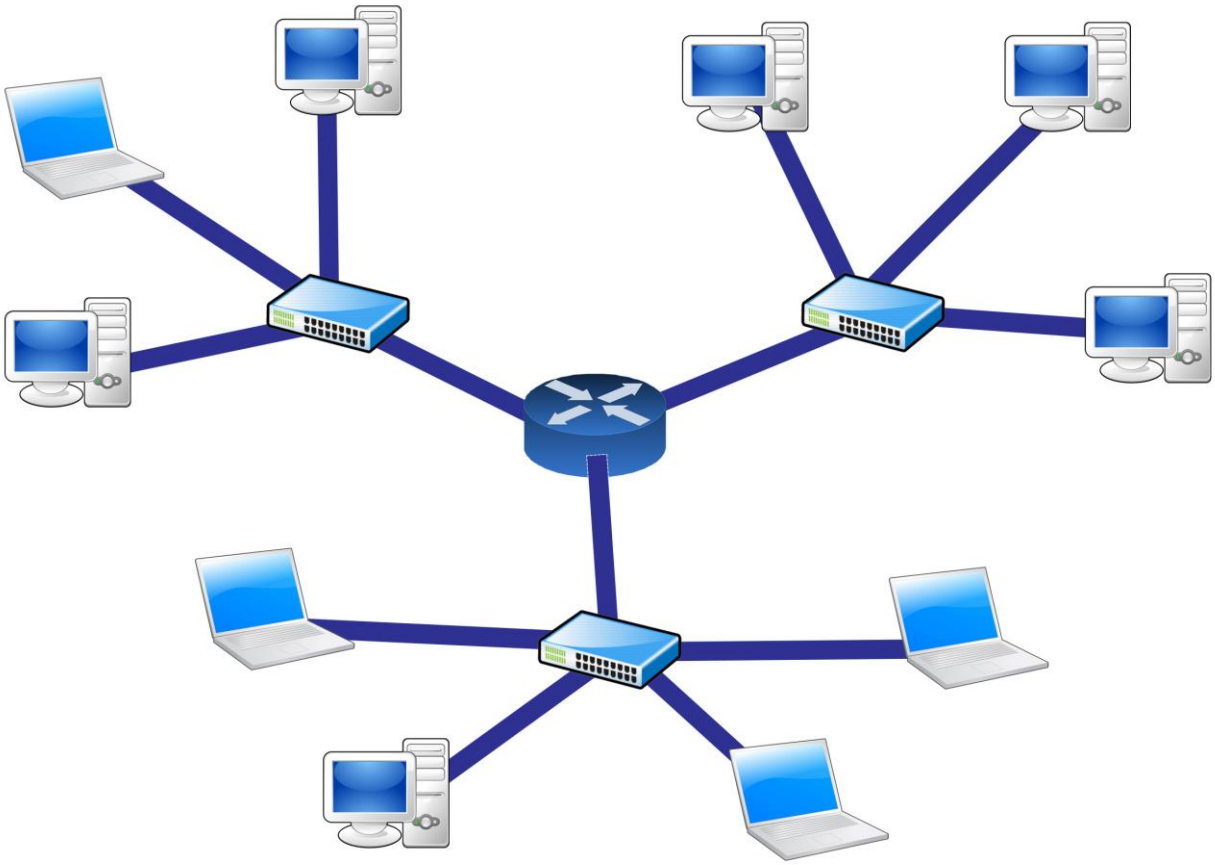
2.1 L'architettura di rete

Prima di entrare nel merito della struttura di una rete "Software-Defined", è necessario esporre alcune caratteristiche essenziali della rete tradizionale odierna. Una rete è composta da un insieme di nodi connessi tra loro tramite collegamenti o link. Tali nodi possono essere, ad esempio, host: per la maggior parte sistemi terminali generici gestiti da utenti come computer, tablet o smartphone (*client*), oppure macchine alquanto capienti (*server*) contenenti grandi quantità di dati e informazioni per l'usufrutto dei client. Per permettere la comunicazione tra nodi non connessi direttamente, sono stati introdotti dei dispositivi di instradamento e commutazione quali switch e router, che sostanzialmente svolgono il compito di recapitare i dati attraverso la rete secondo percorsi ottimali. La differenza lampante tra questi due elementi riguarda per la maggior parte il livello del modello ISO/OSI in cui essi elaborano i pacchetti di dati: mentre gli switch li coordinano solo sul livello di collegamento, i router salgono fino allo strato superiore, ovvero il livello di rete. Le decisioni di instradamento o inoltro di un pacchetto vengono intraprese esclusivamente e autonomamente da ogni dispositivo di commutazione che riceve in ingresso quest'ultimo. Infatti ognuno di questi nodi di commutazione, in particolare lo switch, possiede un piano di controllo o control-plane, e un piano dati o data-plane, che rispettivamente costituiscono il livello funzionale logico e il livello hardware fisico. Il piano di controllo, che si può definire come l'unità logica del sistema, viene generalmente implementata con software proprietario sviluppato direttamente dal produttore dell'apparecchio stesso. Anche questo fattore, come lo stretto appaiamento tra i due piani appena citati, rappresenta un ostacolo non banale davanti allo sviluppo di funzioni o protocolli interamente nuovi. Come già citato nel capitolo precedente, il fenomeno dell'ossificazione di rete rappresenta la staticità della struttura attuale delle reti davanti a cambiamenti come questi ultimi, oppure di fronte al mutamento del traffico moderno, dove la concezione gerarchica della forma client-server sta diventando man mano obsoleta.



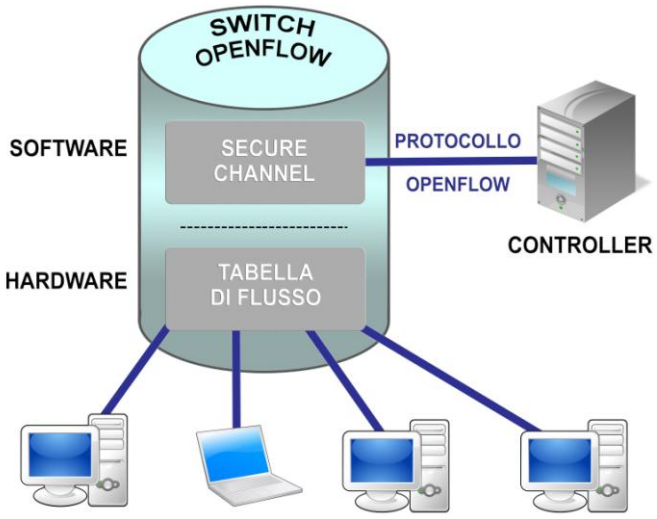
Finora sono stati sviluppate e implementate alcune soluzioni parziali che “bypassano” l’effetto di ossificazione come i middelbox (es. NAT, firewall, ecc..), cioè nodi intermediari che trasformano e adattano i pacchetti in transito secondo le esigenze specifiche degli utenti. Anch’essi vengono rilasciati con un software “chiuso” appartenente al produttore e offrono una flessibilità leggermente maggiore al costo di un aumento della complessità e quindi della latenza per quanto riguarda la trasmissione dei pacchetti nella rete.

In una rete basata su SDN, il piano di controllo viene scisso dal piano di dati e spostato su un ulteriore dispositivo chiamato controller, incaricato appunto di fornire quei servizi e quelle regole pianificati in precedenza dagli switch stessi. Ciò permette non solo una gestione ottimale della traffico di rete indipendentemente dalla posizione fisica dei nodi in essa, ma contribuisce inoltre a risolvere il problema del cosiddetto fenomeno di ossificazione della rete. Trasformare i dispositivi di commutazione come gli switch in puro hardware di inoltro tramite l’astrazione del piano di controllo non solo consente di avere una visione semplice e schematica della rete, ma permette anche lo sviluppo e la distribuzione di nuove applicazioni e protocolli, mutando pertanto i nodi intermedi come i middlebox in un unico software di controllo.



2.2 Gli Switch

In una struttura di rete classica gli switch, o commutatori, sono dispositivi operanti a livello di collegamento che svolgono il compito di filtrare, inoltrare e instradare pacchetti di dati in transito tra host appartenenti ad un'area locale. Mentre l'operazione di filtraggio consiste nella semplice decisione di scartare o no un pacchetto in arrivo, l'inoltro, o forwarding, comprende la scelta di indirizzare un determinato pacchetto di dati verso la corretta l'interfaccia ove quest'ultimo è diretto. L'instradamento infine descrive l'invio di tale pacchetto nella rete tenendo conto dell'indirizzo fisico del nodo destinatario. Per poter effettuare simili operazioni, ogni switch è dotato di una tabella di inoltro, o forwarding table, contenente gli indirizzi MAC dei nodi conosciuti associati alle porte delle interfacce di destinazione che indirizzano ad essi. In una rete tradizionale questa tabella, inizialmente vuota, viene arricchita di voci dallo stesso switch man mano che riceve pacchetti dai diversi host connessi. In altre parole, quando lo switch riceve un pacchetto possono verificarsi due scenari possibili: (1) l'indirizzo di destinazione del pacchetto è già presente tra le voci nella forwarding table, quindi lo switch non fa altro che convogliarlo sull'interfaccia d'uscita associata; (2) l'indirizzo non è contenuto nella tabella, perciò lo switch ne inoltra una copia su tutte le porte attendendo una risposta dal destinatario corretto affinché possa infine registrare l'interfaccia valida da associare all'indirizzo.[5] Questo meccanismo rappresenta al meglio il concetto di controllo decentralizzato di una rete: ogni switch compie decisioni di inoltro dei pacchetti secondo la propria tabella e quindi indipendentemente dal comportamento di altri dispositivi di commutazione o instradamento, il che implica che le scelte riguardanti il flusso di dati attraverso la rete vengono intraprese da ogni singolo elemento di rete.



In una rete basata su Software-Defined Networking invece, uno switch viene ridotto a mero hardware di inoltro accessibile mediante un'interfaccia aperta, in quanto il controllo logico e gli algoritmi di instradamento vengono delegati a un controller. Come già affermato in precedenza, protocolli di rete SDN come OpenFlow identificano il traffico di dati in flusso, pertanto uno switch OF sarà composto da una o più tabelle di flusso, o flow table, che immagazzinano voci per il corretto inoltro e instradamento dei pacchetti, e da un livello di astrazione che comunica, mediante un canale sicuro, con un controller, nel caso in cui alcuni elementi non vengano trovati in queste tabelle. Ognuna di queste tabelle contiene: (1) *campi di corrispondenza*, con informazioni ottenute dall'intestazione (*header*) del pacchetto, porta di ingresso e metadati, usati per identificare i pacchetti in arrivo, (2) *contatori*, utilizzati per memorizzare statistiche per il

[5] Per un approfondimento si veda James F. Kurose, K. Ross: *Reti di Calcolatori e Internet*, Pearson 2007, pp. 405-406

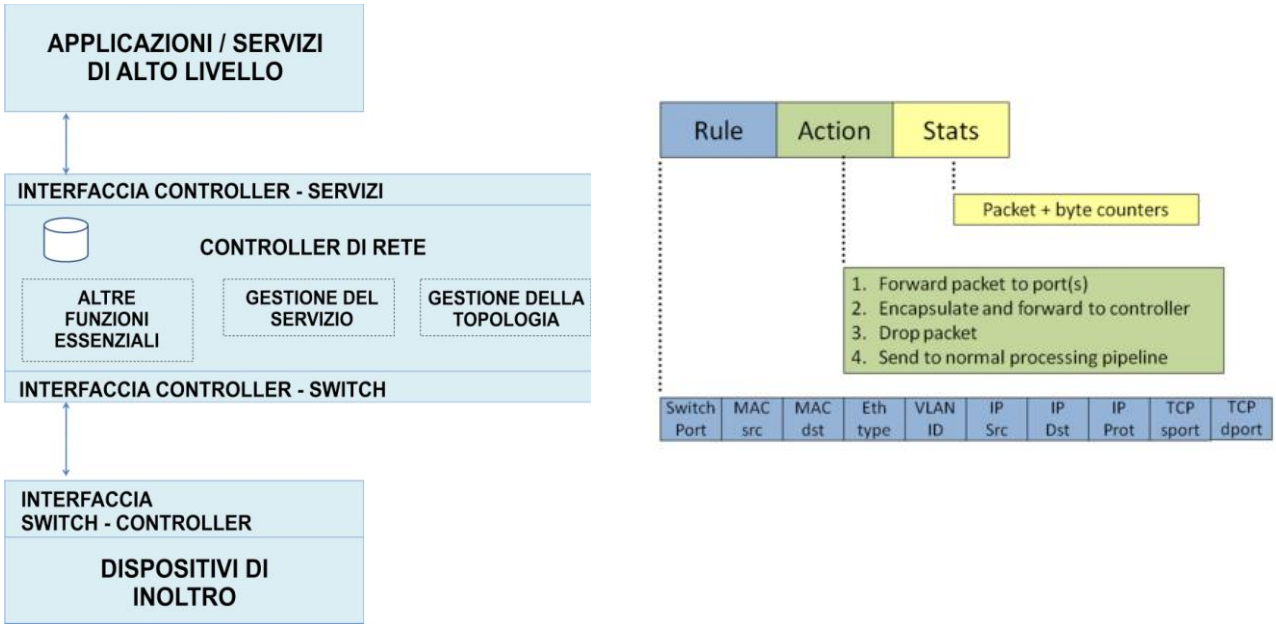
flusso specifico quali numero di pacchetti ricevuti, quantità di byte e durata del flusso di dati, e (3) *un insieme di istruzioni, o azioni* da intraprendere in relazione ad una corrispondenza, che detta come gestire pacchetti correlati.

In una rete con tecnologia OpenFlow, gli switch possono appartenere a due tipologie: puri o ibridi. Uno switch OF puro non ha funzionalità ereditarie o poteri di controllo, perciò fa completo affidamento al controller per le decisioni di inoltro. Uno switch ibrido supporta OF in aggiunta alle operazioni e ai protocolli tradizionali. La maggior parte degli switch disponibili in commerciò è ibrida. Come già detto in precedenza, generalmente, uno switch OF consiste in uno o più tabelle di flusso che immagazzinano voci di flusso per l'instradamento e l'inoltro dei pacchetti. [6]

[6] Mendonca, Astuto, Nunes, Nguyen, Obraczka, Turletti: *A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks*, 2013, pp. 4.

2.3 Il Controller

L'architettura di una rete Software-Defined Networking, rispetto a una rete tradizionale, è dotata di un elemento aggiuntivo chiamato controller. Se una rete locale classica è composta da routers e switches le cui unità di controllo sono integrate ad essi con interfacce specifiche sviluppate dai produttori, quindi con possibilità limitate di rinnovarsi verso nuovi protocolli o servizi di rete, l'aggiunta di un controller e quindi di un software di controllo non solo rende possibile tutto ciò, ma, come già descritto in precedenza, permette una gestione centralizzata dei dispositivi di rete, rendendo questi ultimi pressoché mero hardware. Il funzionamento di questa architettura è stato metaforicamente accostato a quello di un sistema operativo[7] e il controller rappresenta il fulcro di tale meccanismo. Esso permette la comunicazione tra i dispositivi di rete al livello inferiore (Control to Data-Plane Interfaces o Southbound Interfaces) e le applicazioni software a livello superiore (Northbound Interfaces). Ciò consente di percepire una rete come un unico sistema centralizzato coordinato dal controller e personalizzato mediante applicazioni utente, indipendentemente dalla topologia fisica di quest'ultima. Quando un pacchetto raggiunge uno switch OF, le intestazioni vengono confrontate con le voci presenti nella tabella di flusso. Se non viene trovato nessun riscontro, il pacchetto verrà incapsulato e inviato verso il controller. Quest'ultimo perciò avrà pieni poteri sulla gestione del pacchetto di dati, comunicando allo switch interessato se scartarlo, o inoltrarlo inserendo una nuova voce nella tabella.



Naturalmente, il dubbio che può insorgere spontaneamente riguarda il rischio elevato dovuto ad un accentramento di tutte le responsabilità verso un unico dispositivo. Ciò può comportare notevoli limitazioni dal punto di vista dei tempi di risposta, della scalabilità e in particolare dell'operatività; in altre parole, un semplice guasto del controller comprometterebbe inevitabilmente l'intero traffico di dati. Questo porta alla decisione di piazzare su una rete diversi elementi di controllo equivalenti e comunicanti tra loro pur mantenendo, nel complesso, un piano di controllo logico centralizzato. Di fatto, la questione riguardante il piazzamento del controller è ancora in fase di discussione, ovvero esistono diverse teorie riguardo la scelta

[7] N. Gude, T. Koponen, J. Pettit, B. Pfa_, M. Casado, N. McKeown, S. Shenker: *Nox: towards an operating system for networks*, ACM SIGCOMM Computer Communication Review, 2008, pp:105-110.

del numero di controller attivi su una determinata area di rete e in relazione alla loro posizione nella topologia di questa, affinché si possa ottimizzare le performance nei tempi di risposta e garantire affidabilità.

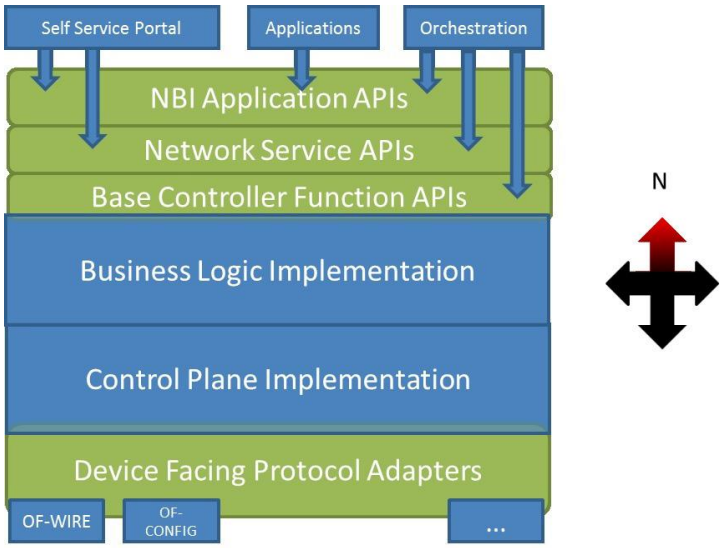
Alcuni metodi e scelte progettuali di SDN infatti permettono strutture che uniscono aspetti dei sistemi decentralizzati senza entrare in conflitto con il concetto di controllo centralizzato. Ad esempio il sistema Kandoo[8] propone una distinzione tra controller locali e globali, dove i primi, non comunicanti tra loro e all'oscuro della topologia di rete, gestiscono il flusso di dati relativo ad applicazioni locali e cioè potenzialmente connettibili mediante un solo switch, mentre i secondi fungono da controller centralizzati convenzionali. Tale sistema permette di scremare il numero di nuove richieste di flusso, scegliendo inoltre il percorso di rete con minori tempi di risposta e di overhead per richieste che possono essere gestite in un contesto locale, isolando il rischio del fenomeno del "collo di bottiglia" esclusivamente nei confronti dei controller globali, preservandone comunque la scalabilità.

Uno studio più recente su diverse implementazioni di controller OpenFlow (NOX-MT, Maestro, Beacon[9]), condotto su una simulazione di rete allargata a 100'000 nodi e fino a 256 switch, ha dimostrato che tutte queste erano in grado di gestire almeno altre 50'000 richieste di flusso al secondo in ciascuno degli scenari testati. In una macchina con CPU 8-core, l'implementazione NOX-MT supportante multi-threading ha gestito 1.6 milioni di nuove richieste di flusso per secondo con un di tempo di risposta medio di 2 millisecondi. Come mostrano i risultati, un singolo controller è in grado di gestire un numero sorprendente di nuove richieste, e potrebbe essere capace di gestire la maggior parte delle reti più estese. Inoltre, nuovi controller in fase di sviluppo come MCNETTLE[10] sono destinati a server multi-core altamente performanti e sono progettati per scalare fino a livelli smisurati di trasmissioni di dati: 20 milioni di richieste al secondo e fino a 5000 switch. Ciononostante, controller multipli possono essere impiegati per trovare il compromesso ideale tra tempi di overhead e affidabilità.[11]

[8] www.kandoo.org/
[9] A. Tootoonchian, S. Gorbunov, Y. Ganjali: *On Controller Performance in Software-Defined Networks*, USENIX conference, 2012.
[10] A. Voellmy, J. Wang: *Scalable Software Defined Network Controllers*, SIGCOMM, 2012.
[11] Per approfondire si veda: *A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks*, 2013, pp. 4-6.

2.4 Interfacce di Livello Superiore (Northbound Interfaces)

Quando i primi controller SDN furono ideati, si pensò a distinguere i componenti collegati superiormente e inferiormente ad esso basandosi rispettivamente sull'orientamento delle frecce nei punti cardinali Nord e Sud. Perciò, le cosiddette Northbound Interfaces (NBI) rappresentano quelle interfacce di programmazione indispensabili per la comunicazione tra i controller SDN e i software applicativi di controllo, inoltre è necessaria un'interazione multilaterale tra le due parti, in quanto determinate funzioni o servizi potrebbero dover acquisire informazioni riguardo i livelli inferiori della struttura di rete, o a proposito del comportamento di questa, oltre che alla normale gestione delle risorse e del traffico tra i vari nodi interconnessi.



Mentre l'implementazione standard delle interfacce inferiori, situate tra controller e switch, è OpenFlow, per quanto riguarda le NBI attualmente non esiste un modello di API dominante. Questa mancanza è dovuta al fatto che, pensando a un controller come un "sistema operativo di rete", già citato in precedenza, è senz'altro necessaria la presenza di un'interfaccia che permetta la comunicazione tra l'unità logica e l'hardware sottostante come un dispositivo di commutazione, ma non è essenziale, per chi sviluppa applicazioni software, la conoscenza dettagliata dell'implementazione di un controller.

Nonostante ciò, la Open Networking Foundation ha recentemente reclutato un team di lavoro che si sta occupando della standardizzazione di un'interfaccia di programmazione per l'interazione controller-applicazioni.[12] Entro la fine del 2014 saranno disponibili il modello e l'implementazione di due casi d'uso.

[12]Per un approfondimento si veda: S. Raza, D. Lenrow: *North Bound Interface*, Working Group (NBI-WG) Charter, 2013.

2.5 Interfacce di Livello Inferiore (Southbound Interfaces)

Come già accennato nel paragrafo precedente, le interfacce chiamate Southbound Interfaces o Control to Data-Plane Interfaces (CDPI), instaurano una comunicazione tra i controller e i dispositivi di commutazione, in altre parole collegano il piano di controllo al piano di dati. In questo caso, l'implementazione standard di interfaccia è proprio il protocollo OpenFlow. Questo protocollo può essere paragonato all'insieme di istruzioni di un processore, cioè la successione di quelle variabili primitive che possono essere visibili agli sviluppatori del software.[13] Esso viene implementato sia sul lato del controller che sul lato dei dispositivi e instaura così in canale di comunicazione end-to-end sicuro grazie ai convenzionali protocolli criptografici come SSL, o TLS.

Dunque questa interfaccia rappresenta un punto critico dell'architettura SDN, in quanto non solo permette al controller di gestire in maniera dinamica e atomica il traffico di una rete, ma ne incrementa l'efficienza in termini di traffico e richieste di transito.

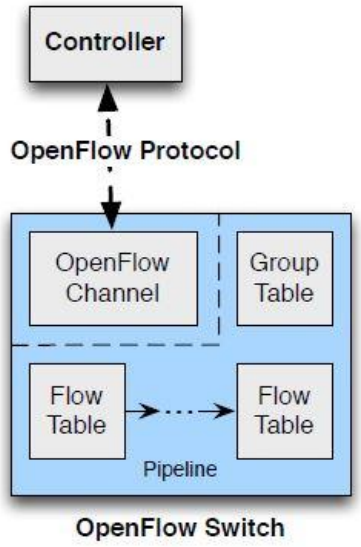
Nel capitolo successivo verranno messi a fuoco proprio gli aspetti peculiari e il funzionamento del protocollo OpenFlow implementato su questa interfaccia di comunicazione controller-switch.

[13] ONF White Paper: *Software-Defined Networking: The New Norm for Networks*, Open Networking Foundation, 2012, pp: 9.

3. OPENFLOW

3.1 Descrizione del protocollo OpenFlow

Uno switch compatibile con OpenFlow contiene generalmente una o più tabelle di flusso (*flow table*) e una tabella di gruppo (*group table*) i quali realizzano il controllo e l'inoltro dei pacchetti, e infine uno più canali OF (*OpenFlow channel*) di comunicazione con il controller. Mediante il protocollo OF, il controller può aggiungere, modificare e cancellare le voci della tabella di flusso sia in maniera reattiva che proattiva.[14]



Mediante il canale OpenFlow (*OF channel*), l'interfaccia che crea una connessione diretta tra switch e controller, quest'ultimo configura e gestisce in modo da permettere lo scambio di messaggi e l'invio o ricezione di pacchetti. Il canale di controllo (*control channel*) di uno switch può supportare uno o più canali OF secondo il numero di controller connessi a esso.

3.1.1 Messaggistica del Protocollo

Per quanto riguarda i messaggi del canale controller-switch, il protocollo OpenFlow prevede un modello standard da applicare su ogni dispositivo rilasciato. Sono supportate tre tipologie di messaggio: controller-verso-switch (*controller-to-switch*), asincrono (*asynchoronous*) e simmetrico (*symmetric*).

- I messaggi **controller-to-switch** sono sempre inviati inizialmente dal controller e di norma non è richiesta necessariamente una risposta da parte dello switch. La maggior parte di questi messaggi può avere lo scopo di interrogare il dispositivo di commutazione riguardo alle proprie capacità di base (*features*) e specifiche tecniche (*read-state*), di modificare alcune impostazioni (*configuration*)

[14] *OpenFlow Switch Specification*, The Open Networking Foundation, 2014

o determinate voci nelle tabelle di flusso (*modify-state*), e infine di inviare e inoltrare pacchetti di dati ricevuti in precedenza per essere esaminati (*packet-out*).

- I messaggi **asincroni** appartengono alla categoria "push", in altre parole vengono inviati dagli switch ai controller senza che questo ne solleciti l'invio. La circostanza più importante riguarda l'invio al controller di un pacchetto per il quale non è stata trovata alcuna corrispondenza nelle tabelle di flusso (*packet-in*), ma questo genere di messaggio può comprendere anche alcuni tipi di notifiche finalizzate a informare il controller di alcune modifiche effettuate dallo switch quali la rimozione di una voce dalla tabella di flusso (*flow-removed*), il cambio di una porta (*port-status*) o semplicemente il manifestarsi di un problema (*error*).

- I messaggi **simmetrici** sono bilaterali e vengono inviati senza sollecitazione. Essi comprendono messaggi scambiati al fine di inizializzare la connessione tra i due dispositivi (*hello*) o semplici verifiche che tale connessione sia ancora attiva (*echo*).

3.1.2 Il Canale OpenFlow

Un controller OpenFlow è in grado di gestire più canali OF, ognuno proveniente da uno switch, mentre questi può avere un canale indirizzato a un singolo controller oppure multipli canali per ogni controller. Generalmente queste connessioni del canale OF vengono inizializzate da parte dello switch mediante il protocollo criptografico TLS. Alternativamente, il controller può instaurare la connessione se lo switch ha una porta TCP riservata.

In caso di switch gestiti da due o più controller verrà assegnato a questi ultimi un ruolo definito in modo da ottimizzare l'amministrazione del traffico di rete. Un controller può avere pieni poteri e ricevere tutte le informazioni dai messaggi asincroni se configurato per l'assegnazione dei ruoli "master" o "equal", dove il primo può essere conferito al più a un dispositivo, mentre il secondo non ha tale limitazione ed è il ruolo di default. Il ruolo restante, "slave", imposta il controller in modalità di sola lettura e filtra una parte dei messaggi asincroni inviati dallo switch. La presenza di controller multipli ottimizza la stabilità e l'affidabilità di una rete, prevenendo situazioni critiche come la perdita di connessione con uno di essi.

Infine, oltre alla suddetta connessione principale "controller-switch", un canale OF può essere composto da ulteriori connessioni ausiliarie utili per migliorare le performance di elaborazione poiché favoriscono il parallelismo implementato sulla maggior parte degli switch prodotti.

3.2 Componenti di uno Switch OpenFlow

Come già accennato in precedenza, ogni elemento della tabella di flusso è composto al più da: campi di corrispondenza (*match fields*), contatori (*counters*), e un insieme d'istruzioni (*instructions*) da eseguire per i pacchetti relativi. Come verrà mostrato successivamente, il controllo inizia dalla prima tabella di flusso e può proseguire fino alle tabelle supplementari della pipeline di processo.

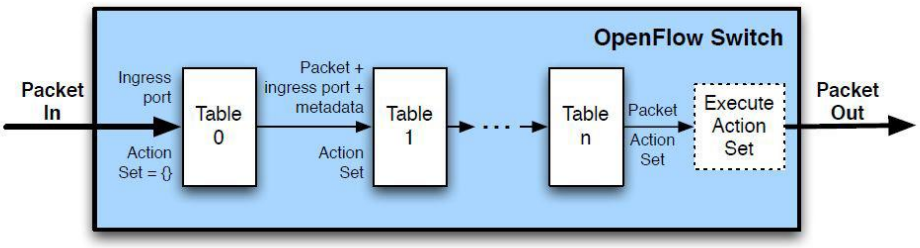
3.2.1 Pipeline

Un pacchetto ricevuto da uno switch OF puro viene conseguentemente elaborato dalla pipeline OpenFlow, ovvero una serie di analisi di confronto con i campi contenuti nelle tabelle di flusso. Uno switch ibrido invece supporta anche le normali operazioni di commutazione ed è in grado di spostare un pacchetto dalla pipeline OF alla procedura classica tramite l'inoltro verso le porte riservate NORMAL e FLOOD.

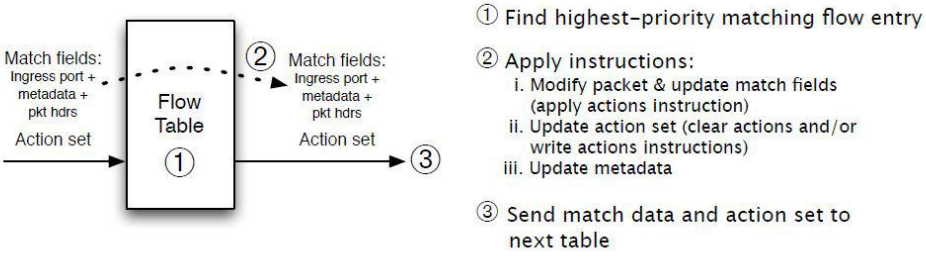
Analogamente ai vettori di un linguaggio di programmazione, le tabelle di flusso contenute in un dispositivo vengono allocate in serie numerica a partire da 0. Il processo di confronto tra un dato pacchetto e l'insieme delle tabelle di flusso avverrà esclusivamente in direzione ascendente, pertanto inizierà sempre dalle voci presenti nella tabella numero 0.

Se viene trovata una corrispondenza su un elemento, la ricerca terminerà e, in seguito all'aggiornamento del contatore concernente i riscontri della voce confrontata, sarà eseguito l'insieme d'istruzioni presenti in quest'ultima. In alcuni casi possono essere presenti istruzioni chiamate "Goto-Table" che, come un puntatore, indirizzano il pacchetto a un'ulteriore tabella di flusso, a condizione che il numero della tabella di destinazione sia maggiore del numero della tabella di partenza.

Se invece la ricerca non dà risultati, il pacchetto verrà associato alla voce table-miss, o mancanza di tabella, che definisce come gestire i pacchetti in caso d'incongruenze e generalmente contiene istruzioni quali l'invio al controller o lo scarto di questi ultimi. Se invece la voce table-miss dovesse mancare, i pacchetti verranno eliminati automaticamente.



(a) I pacchetti vengono confrontati attraverso le tabelle della pipeline



(b) Elaborazione per ogni tabella

3.2.2 Tabella di Flusso e Tabella di Gruppo

Una singola voce contenuta in una tabella di flusso avrà i seguenti valori:

Match Fields	Priority	Counters	Instructions	Timeouts	Cookie	Flags

- **Campi di corrispondenza (*match fields*)**: sono i valori necessari al confronto con i pacchetti di dati e comprendono generalmente la porta d'ingresso, l'intestazione di pacchetto e, in caso di pipeline con due o più tabelle, eventualmente altri metadati relativi a tabelle precedenti;
- **Contatori (*counters*)**: vengono aggiornati in caso di riscontro;
- **Priorità (*priority*)**: valore numerico assegnato in caso di due o più corrispondenze per pacchetto;
- **Istruzioni (*instructions*)**: l'insieme d'istruzioni da eseguire in caso di uguaglianza con un pacchetto;
- **Time out**: valore numerico che definisce il lasso di tempo che precede l'eliminazione della voce;
- **Cookie**: insieme di valori utilizzati dal controller per estrapolare informazioni statistiche relative alla voce, pertanto non vengono considerati durante il processo di pipelining;
- **Bandiere (*flags*)**: utilizzate per differenziare la gestione di una voce;

A differenza della tabella appena descritta, una tabella di gruppo (*group table*) fa riferimento a due o più voci insieme, affinché sia possibile applicare ad esse lo stesso insieme d'istruzioni. Una voce di questa tabella sarà caratterizzata dai seguenti valori:

Group Identifier	Group Type	Counters	Action Buckets

- **Identificatore di gruppo (group identifier)**: valore intero univoco che contraddistingue un determinato gruppo di voci;
- **Semantica di gruppo (group type)**: necessario per compiere una selezione sugli insiemi di azioni da intraprendere nei confronti dei pacchetti;
- **Contatori (counters)**: vengono gestiti in maniera identica ai contatori delle tabelle di flusso;
- **Insiemi di azioni (action bucket)**: una lista di insiemi di azioni, dove ognuna di esse comprende un insieme di istruzioni da applicare assieme ai parametri relativi;

3.2.3 Porte OpenFlow

Le porte OpenFlow non sono altro che le interfacce necessarie per il transito dei pacchetti a partire dall'elaborazione OF fino al resto della rete. Gli switch OF sono comunicanti tra loro tramite le proprie porte di ingresso e uscita, perciò solo mediante quest'ultime sarà possibile inviare un pacchetto rispettivamente da un commutatore OF a un altro. Per consentire l'elaborazione dei pacchetti, uno switch OF alloca un

determinato numero di porte disponibili. Questo insieme di porte OF potrebbe differire dall'insieme di interfacce di rete fornite dall'hardware dello switch, alcune di queste possono non essere disponibili per OpenFlow e, viceversa, alcuni switch possono includere porte OF aggiuntive.

I pacchetti in arrivo vengono accettati attraverso una porta di ingresso e procedono all'esecuzione della pipeline OpenFlow, che in genere li conduce verso una porta d'uscita. Tale pacchetto mantiene il possesso della porta d'ingresso per l'intera durata della pipeline, dopo la quale verrà deciso se inoltrarlo verso una porta di uscita tramite l'azione di output, che definisce in che modo il pacchetto verrà reimmesso nella rete.

Uno switch OF supporta tre tipologie di porte:

a) **Porte Fisiche**: porte che corrispondono ad interfacce hardware appartenenti allo switch. In alcuni casi possono essere virtualizzate attraverso l'hardware di commutazione, rappresentando di conseguenza solo una parte virtuale dell'interfaccia di quest'ultimo;

b) **Porte Logiche**: sono porte definite dallo switch che non corrispondono direttamente a un'interfaccia hardware di esso. Le porte logiche sono astrazioni a livello superiore che possono essere definite mediante metodi diversi da OF, come le interfacce di loopback, link aggregation o tunneling, e possono comprendere l'incapsulamento di pacchetti e il port mapping su una o più porte fisiche. L'elaborazione eseguita da tali porte deve essere dipendente all'elaborazione OF a livello di implementazione, in modo che le porte logiche possano essere trattate come porte fisiche. L'unica differenza tra porte fisiche e logiche sta in una voce aggiuntiva chiamata "Tunnel-ID", presente su un pacchetto processato a livello logico e contenente l'indirizzo sia della porta logica sia della porta fisica associata;

c) **Porte Riservate**: definiscono azioni di inoltro generiche come l'invio di pacchetti verso un controller, il flooding, o l'inoltro tradizionale senza metodi OpenFlow.

- **ALL**: rappresenta tutte le porte che lo switch può usare per inoltrare un determinato pacchetto, può essere utilizzata solo come porta d'uscita. Una copia del pacchetto verrà inviata su tutte le porte standard OF, ad eccezione della porta d'ingresso di quest'ultimo e delle porte configurate ad hoc;

- **CONTROLLER**: rappresenta il canale di controllo con i controller OF e può essere usata sia come porta d'ingresso che d'uscita. Nel primo caso il pacchetto viene identificato come "ricevuto dal controller", mentre nel secondo caso viene incapsulato e inviato mediante il protocollo OpenFlow;

- **TABLE**: utilizzabile solo come porta in uscita, conduce il pacchetto verso l'inizio della fase di ricerca nelle tabelle di flusso;

- **IN_PORT**: invia il pacchetto attraverso la sua porta di ingresso;

- **ANY**: come il Jolly nel gioco delle carte, viene utilizzata nel caso particolare in cui un'operazione di OpenFlow può essere eseguita su una porta qualsiasi;

- **LOCAL**: permette ad elementi remoti l'interazione con lo switch mediante la rete OF;

- **NORMAL**: consiste nell'inoltro tradizionale senza OpenFlow;

- **FLOOD**: rappresenta il flooding classico effettuato dagli switch non-OpenFlow;

Mentre le prime cinque porte sono indispensabili per uno switch OpenFlow qualsiasi, le ultime tre sono presenti solo nei commutatori ibridi.

3.3 Controller basati su OpenFlow

Come già visto in precedenza, per poter gestire le tabelle di flusso di uno switch tramite software compatibile e quindi connesso con quest'ultimo, è necessaria la presenza di un dispositivo equivalente ad un sistema operativo che funga da intermediario tra lo switch e i suddetti programmi utente. Il controller esplica tali funzioni e sono state prodotti diversi modelli che realizzano tutto ciò servendosi del protocollo OpenFlow.

3.3.1 NOX

Il controller NOX, sviluppato dalla Nicira Networks, è stato il primo controller SDN in grado di supportare il protocollo OpenFlow ed è stato donato alla comunità nel 2008 per fornire una base su cui iniziare vari progetti di ricerca. NOX è dotato di un'interfaccia programmabile per OpenFlow scritta nel linguaggio C++, mentre POX, versione modificata e più recente di NOX, è implementata con Python. Entrambi i controller sono orientati verso sistemi Linux.[15]

3.3.2 Beacon

Una valida alternativa NOX si chiama Beacon ed è stato creato dalla Stanford University all'inizio del 2010. Scritto in codice Java, quindi provvisto di un'elevata portabilità, ha dimostrato inoltre un'eccellente stabilità durante i test di gestione di rete. Ha una licenza open source ed è caratterizzato da un rapido sviluppo, grazie alla possibilità di modificare e aggiornare parti di codice senza interromperne l'esecuzione.[16]

3.3.3 Altri Controller OpenFlow

- **Maestro**: rilasciato con licenza open source dalla Rice University, è un controller con API OpenFlow in Java meno comune dei due precedenti;[17]
- **Floodlight**: anch'esso scritto in linguaggio Java in quanto derivato da Beacon, è stato sviluppato da Big Switch Networks per un utilizzo orientato alle imprese piuttosto che alla ricerca;[18]
- **Trema**: scritto in C e Ruby dalla NEC, supporta solo sistemi Linux.[19]

[15] http://www.noxrepo.org/
[16] https://openflow.stanford.edu/display/Beacon/Home
[17] Z. Cai A. Cox T. Ng: Maestro: *A System for Scalable OpenFlow Control*, Rice University, 2010
[18] http://www.projectfloodlight.org/floodlight/
[19] https://www.sdncentral.com/projects/trema/

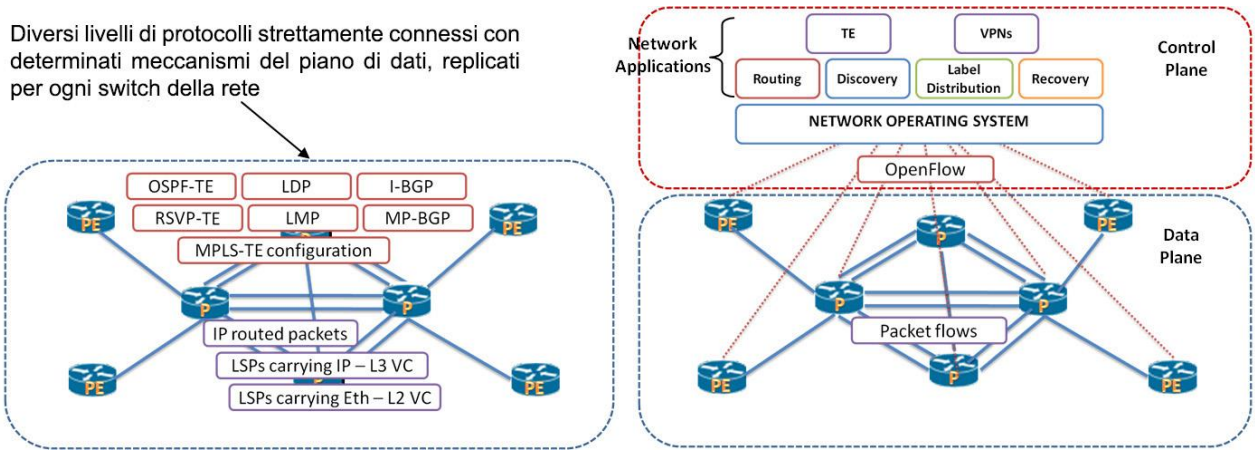
3.4 MPLS vs SDN

Il Multiprotocol Label Switching è una tecnologia sviluppata dalla Internet Engineering Task Force (*IETF*) intorno la seconda metà degli anni '90 che permette di distribuire equamente il traffico di rete tramite percorsi di rete alternativi (*traffic engineering*) operando tra il livello di collegamento e il livello di rete con lo scopo di ottenere una gestione del traffico alquanto efficiente. Tale tecnologia fornisce inoltre metodi per instaurare connessioni private VPN sui livelli citati sopra. Tutto ciò è possibile grazie alla presenza di un'etichetta chiamata MPLS Header, contenuta tra le intestazioni del terzo e del secondo livello di un pacchetto e assegnata ad esso da un Label Switch Router (*LSR*), responsabile inoltre della commutazione di questi. Progettato per garantire la qualità del servizio a differenza del protocollo IP tradizionale, anche MPLS, come l'architettura SDN, è caratterizzato da un piano di controllo unificato e distinto dal piano di dati.

L'adozione di questa tecnologia può portare diversi benefici come l'aumento della performance di inoltro dei pacchetti, la prevenzione dalla congestione di rete, o l'elevata scalabilità. Tuttavia essa presenta alcuni svantaggi considerevoli concernenti ad esempio la vulnerabilità della rete, ma soprattutto l'elevato costo dovuto all'acquisto di router multi-core complessi come i LSR, in sostituzione agli apparati tradizionali non compatibili con le etichette e i protocolli MPLS. Inoltre, con l'avvento di nuove tecnologie, il piano di controllo dei sistemi MPLS sta diventando sempre più complesso a causa dell'inserimento di ulteriori protocolli sviluppati per gestire il piano di dati associato.

Nel 2011 è stato proposto un approccio rivoluzionario che suggerisce la coesistenza tra due tecnologie sostitutive quali MPLS e SDN. [20] Si tratta di una rete MPLS standard con il piano di controllo estensibile basato su OpenFlow-SDN e quindi disaccoppiato dai dispositivi di commutazione, nei quali non sarà necessario innestare i vari strati di protocolli. Oltre ad accentrare in un unico piano di controllo tutti i servizi e le funzionalità dei sistemi basati su MPLS, grazie al protocollo OpenFlow sarà possibile ottimizzare questi ultimi rendendoli ancora più dinamici e facilmente programmabili per l'implementazione di nuove caratteristiche.[21]

Da Giugno 2011, con il rilascio della versione 1.1, OpenFlow è perfettamente in grado di supportare la lettura e l'elaborazione delle etichette MPLS.



[20] Per un approfondimento si veda S. Das, A. R. Sharafat: *MPLS with a Simple OPEN Control Plane*, Stanford University, 2011
[21] *MPLS with OpenFlow / SDN*, (http://archive.openflow.org/wk/index.php/MPLS_with_OpenFlow/SDN)

4 STRUMENTI DI SVILUPPO BASATI SU SDN / OPENFLOW

4.1 Applicazioni basate su OpenFlow

4.1.1 Gestione e configurazione della rete

Come già visto, un'applicazione basata sul paradigma SDN e OpenFlow può semplificare e automatizzare la gestione di una rete. Il rilascio di questa tecnologia ha incentivato la ricerca e implementazione di software applicativo programmato per consentire una gestione di rete dinamica e intuitiva.

Dal 2011 il gruppo di ricerca dell'Università di Rio de Janeiro ha presentato il tool "OMNI"[22], attualmente integrato nel software "FITS", che presenta un'interfaccia utente eseguita su web browser ed è compatibile con i controller OpenFlow basati su NOX. OMNI permette di configurare la rete mediante funzioni di monitoraggio e di gestione dinamica dei flussi e inoltre è dotato di un sistema multi-agente che aumenta le perfomance di traffico minimizzando le perdite di pacchetti di dati.

Alla Brown University è stato presentato un metodo[23] basato su OF che introduce le "tabelle di flusso gerarchiche" (*Hierarchical Flow Tables*, *HFT*), cioè un sistema che definisce una struttura ad albero delle regole e politiche delegate a diversi amministratori coesistenti secondo un ordine preciso. Ciò permette di avere un confronto netto tra queste regole di alto livello e tra la topologia fisica della rete in modo da ottimizzare le performance di traffico.

Yamasaki dell'università di Tohoku ha proposto un sistema[24] di gestione delle VLAN mediante tecnologia SDN/OpenFlow, con particolare attenzione agli ambienti accademici. In una rete informatica di un ateneo e del suo ambiente circostante è spesso necessaria una notevole quantità di LAN virtuali affinché sia possibile compiere delle distinzioni tra varie porzioni di nodi e soprattutto realizzare un controllo degli accessi altamente sofisticato. Tuttavia le VLAN, oltre che richiedere una configurazione e una manutenzione laboriose, sono contraddistinte da limitazioni notevoli, come ad esempio il numero massimo di ID assegnabili corrispondente a 4096 (12 bit). Il prototipo testato nei campus nipponici e basato sull'architettura OpenFlow presenta un sistema molto più leggero rispetto alle VLAN tradizionali e soprattutto non possiede alcun limite di accessi, grazie agli ID virtuali generati e assegnati dal controller.

4.1.2 Sicurezza

Nei sistemi OpenFlow possono essere eseguite diverse applicazioni che controllano il flusso di dati allo scopo di prevenire un attacco esterno o modificano il traffico in modo da isolare una particolare porzione della rete.

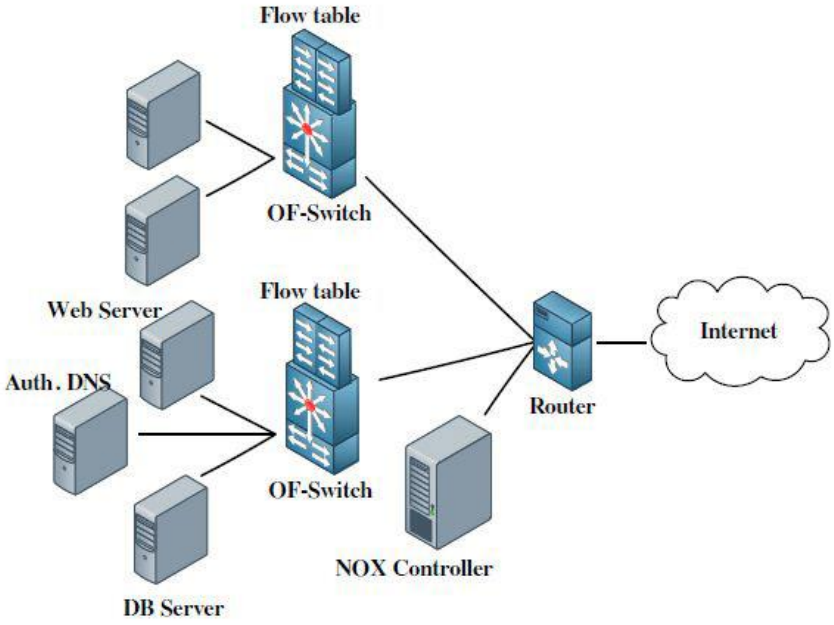
---

[22] M. F. Mattos: *OMNI: OpenFlow MaNagement Infrastructure*, Universidade Federal do Rio de Janeiro, 2011
[23] A. D. Ferguson: *Hierarchical Policies for Software Defined Networks*, Brown University, 2012
[24] Y. Yamasaki: *Flexible Access Management System for Campus VLAN Based on OpenFlow*, 2011 IEEE/IPSJ 11th International Symposium on Applications and the Internet, 2011

Esistono diversi metodi proposti per la rilevazione di attacchi multipli alle risorse e ai servizi di un determinato sistema informatico (*Denial of Service, DDoS*), il più rilevante è sicuramente il "CONA"[25], attualmente implementato sugli switch OpenFlow NetFPGA. Questo sistema etichetta i diversi flussi di traffico secondo l'host richiedente e il contenuto domandato. Nel caso in cui il server superi un numero predefinito di richieste di accesso ricevute per la medesima risorsa, il sistema potrà individuare un attacco di tipo DDoS.

Un team di ricerca dell'università di Pechino invece si propone di compensare l'attuale mancanza di una convalida degli indirizzi IP dei pacchetti inoltrati con l'architettura denominata "VAVE"[26], compatibile con il sistema NOX. Quando uno switch riceve in ingresso un pacchetto che non appare nelle tabelle di flusso, quest'ultimo sarà inviato al controller e l'indirizzo sorgente verrà convalidato. In caso di sospetto attacco informatico, sarà applicata una regola per bloccare il transito.

Il sistema "OpenFlow Random Host Mutation" (*OFRHM*)[27] si basa su una tecnica informatica chiamata "Moving Target Defense" (*MTD*) e protegge la rete da intrusioni malevoli mediante un modifica dinamica degli indirizzi IP dei nodi connessi. In questo caso specifico, un controller OpenFlow assegna frequentemente ad ogni host un indirizzo IP virtuale collegato a quello reale. Tramite DNS sarà possibile raggiungere gli host utilizzando l'IP virtuale, mentre solamente i nodi autorizzati possono raggiungerli con l'indirizzo reale. Ciò previene la rete da "stealth scanning" e tutti le altre tecniche sulla rilevazione celata.



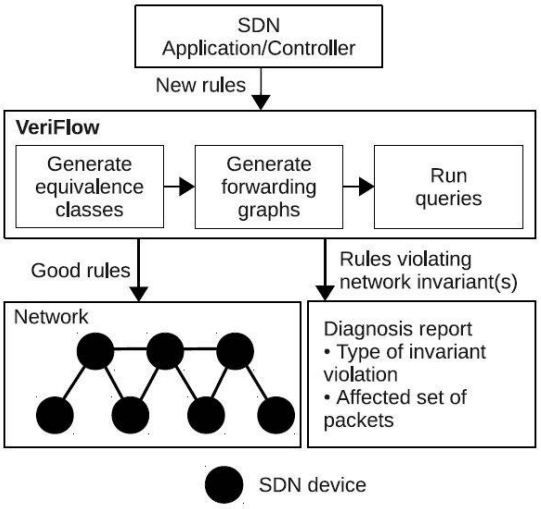
Architettura della rete OF-RHM

[25] J. Suh: *Implementation of Content-oriented Networking Architecture (CONA): A Focus on DDoS Countermeasure, Seoul National University*, 2010
[26] G. Yao: *Source Address Validation Solution with OpenFlow/NOX Architecture*, Tsinghua University, 2011
[27] J. H. Jafarian: *OpenFlow Random Host Mutation: Transparent Moving Target Defense using Software Defined Networking*, University of North Carolina, 2012

4.1.3 Viabilità di rete

"Plug-n-Serve"[28] è uno strumento di bilanciamento del carico (*load balancing*) sviluppato alla Stanford University nel 2009 e successivamente migliorato col nome di "Aster*x". La tecnica del "load balancing" consiste nella distribuzione ponderata dell'elaborazione del traffico tra due o più nodi. Questa applicazione basata su OpenFlow si occupa di ridurre i tempi di risposta in base al carico di lavoro dei server e alla congestione di rete monitorati in tempo reale. Per ottenere dei risultati ottimali, "Plug-n-Serve" modifica i percorsi di instradamento e aggiunge oppure rimuove ulteriori risorse di elaborazione come ad esempio un server, ovviamente mediante l'utilizzo del controller.

Uno degli aspetti fondamentali che condizionano l'affidabilità di una rete riguarda la tolleranza ai guasti (*fault tolerance*), ovvero la capacità di non interrompere il servizio in presenza di errori. "VeriFlow"[29] è un'applicazione che controlla alla ricerca di incongruenze in ogni dispositivo connesso alla rete in tempo reale oppure ogni volta che una voce della tabella di flusso viene inserita, modificata o cancellata, impiegando al più qualche centinaia di millisecondi. Sviluppato per i sistemi NOX, questo strumento supporta l'analisi di intestazioni di pacchetto multiple ed è dotato di un'interfaccia per la ricerca di incoerenze specifiche.



VeriFlow opera tra le applicazioni SDN e i dispositivi per intercettare e controllare ogni regola introdotta nella rete

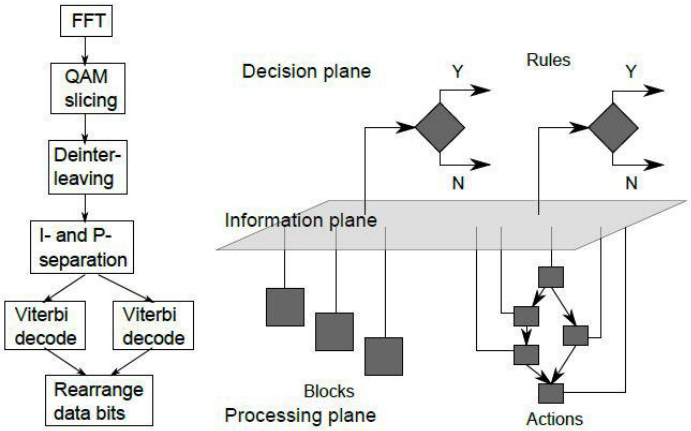
4.1.4 Applicazioni Mobili

Le reti OpenFlow wireless stanno attualmente ottenendo molta considerazione tra i ricercatori accademici, in quanto, come sarà approfondito in seguito, la richiesta odierna di connessioni senza fili è in costante aumento. Infatti, alla Stanford University è stata sviluppata una piattaforma aperta soprannominata "OpenRoads"[30] che permette ai ricercatori di osservare e sperimentare ogni possibile soluzione innovativa per lo sviluppo delle reti mobili. Questa applicazione conferisce il totale controllo dei dispositivi connessi

[28] N. Handigol: *Plug-n-Serve: Load-Balancing Web Traffic using OpenFlow*, Stanford University, 2009
[29] A. Khurshid: *VeriFlow: Verifying Network-Wide Invariants in Real Time*, University of Illinoi, 2012
[30] K. Yap: *OpenRoads: Empowering Research in Mobile Networks*, Stanford University, 2009

mediante lo standard OpenFlow e il protocollo SNMP (*Simple Network Management Protocol*), che a sua volta fornisce un'interfaccia di gestione e configurazione della rete.

Sempre alla Stanford University è stata sviluppata un'infrastruttura wireless basata sul paradigma Software-Defined Networking chiamata "OpenRadio"[31], dotata di un'interfaccia programmabile scomponibile e intuitiva che mira a riscrivere i protocolli tradizionali scindendo piano di elaborazione e piano decisionale. Il piano di elaborazione comprende un insieme di schemi associati a diversi tipi di azioni, mentre il piano decisionale contiene l'unità logica che sceglie quale schema utilizzare per un determinato pacchetto in transito, ad esempio un flusso di dati contenente streaming video comporterà l'esecuzione di una specifico modello algoritmico.



a) Grafo aciclico di un'azione personalizzata per la trasmissione video

b) Modello di OpenRadio con piano di decisione e di elaborazione separati

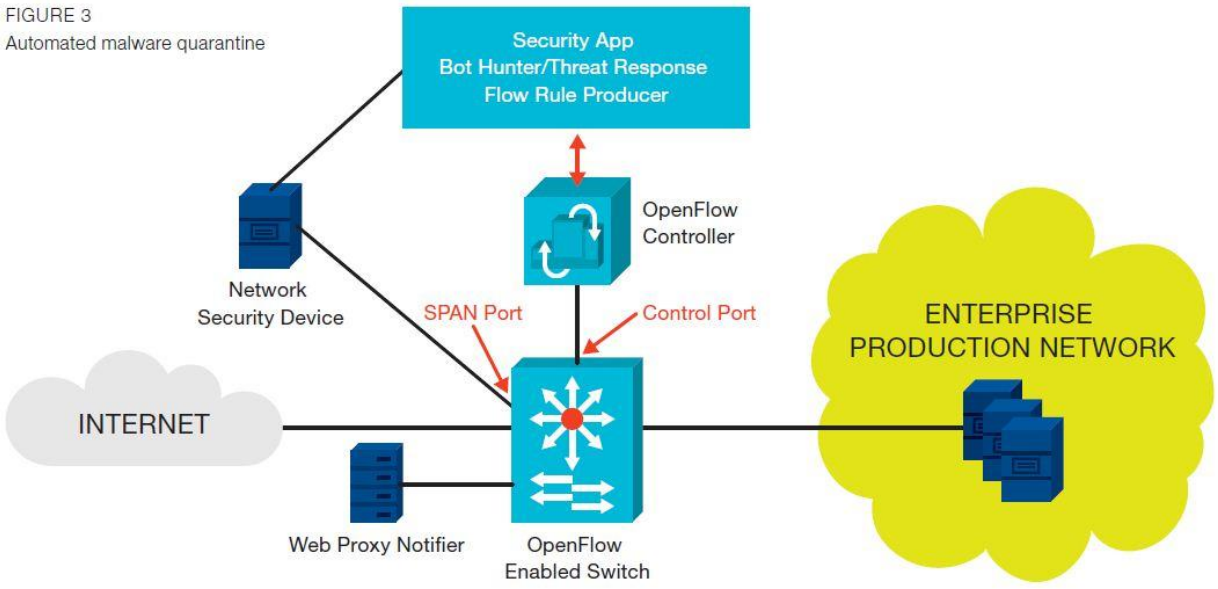
Infine, per quanto riguarda le infrastrutture wireless locali o WLAN, è stato rilasciato "Odin"[32] un prototipo di architettura SDN progettato per le reti aziendali che semplifica la gestione degli host connessi. Tra le funzioni principali di questo sistema sono inclusi autentificazione, autorizzazione e registrazione degli host, gestione delle interferenze e bilanciamento del carico.

[31] M. Bansal: *OpenRadio: A Programmable Wireless Dataplane*, Stanford University, 2012
[32] L. Suresh: *Towards Programmable Enterprise WLANs with Odin*, Proceedings of the first workshop on Hot topics in software defined networks, 2012

4.2 Aree di applicazione

4.2.1 Sicurezza e Data Center

Con la richiesta in aumento dei servizi come il "Cloud Computing" che permette all'utente di poter avere i propri dati a portata di mano in qualsiasi situazione o area geografica, le imprese del settore stanno man mano usufruendo di ampi centri di calcolo detti Data Center mediante l'acquisizione oppure la disponibilità da parte di terzi. I Data Center sono in continua evoluzione verso un ambiente virtuale che soddisfi ogni bisogno del consumatore, compreso l'accesso ai propri dati in qualunque contesto, il tutto con la massima affidabilità.[33]



Uno dei maggiori ostacoli posti davanti a questa evoluzione riguarda proprio la sicurezza dei contenuti immagazzinati nei Data Center. Attualmente esistono numerose politiche di sicurezza situate su ogni livello di rete per prevenire il manifestarsi di minacce. Tra le più impiegate sono inclusi i "Firewall" che fungono da filtri dei pacchetti in transito, gli impianti "Intrusion Detection System" (*IDS*) che controllano e prevengono la rete da attività malevoli, le reti private "Secure Sockets Layer" (*SSL VPNs*) che smistano gli utenti dai domini, le reti ad accesso autentificato mediante porte "IEEE 802.1X", o tutti i sistemi crittografici come "IPsec" per i pacchetti IP e il "Transport Layer Security" (*TLS*) per le connessioni "end-to-end".

Tutte queste soluzioni sono tuttavia difficili da gestire e implementare, oltre che poco inclini alla scalabilità e non interamente affidabili. Inoltre la stretta relazione tra hardware e software rende ancora più ardui gli interventi per fronteggiare le minacce potenziali attraverso attrezzature appartenenti a produttori eterogenei. La tecnologia SDN orientata verso OpenFlow offre una varietà di funzioni che assecondano efficientemente l'evoluzione di servizi offerti dai Data Center. Mentre il controllo logico centralizzato permette un monitoraggio dei pericoli efficace e dinamico attraverso l'intera rete indipendentemente dalla sua struttura fisica, la gestione flessibile e atomica del traffico implica il blocco e quindi l'isolamento degli interventi malevoli senza incidere sulla qualità di navigazione degli utenti connessi.

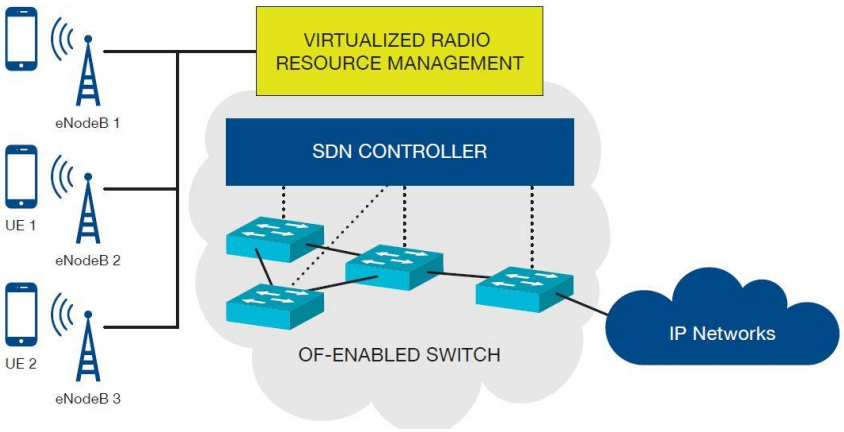
[33] M.McBride: *SDN Security Considerations in the Data Center*, ONF Solution Brief, 2013, p. 4

Un ulteriore problema relativo alla manutenzione di un Data Center riguarda il consumo energetico. La richiesta di servizi comporta un'attività massiccia dei dispositivi sollecitati, in particolare i server, che necessitano sistemi di raffreddamento speciali oltre che un'elevata potenza di calcolo. Infatti è stato stimato che circa il 70% dell'energia destinata a un Data Center viene utilizzata per preservare il corretto funzionamento dei server.[34] I ricercatori della Stanford University hanno presentato uno strumento per l'ottimizzazione del consumo di energia elettrica chiamato "ElasticTree"[35], che regola in modo dinamico il funzionamento dei dispositivi di commutazione attivi nella rete calcolando i percorsi meno dispendiosi in termini di potenza e disattiva invece i nodi non necessari in modo da agevolare e bilanciare il carico di lavoro dei server. I test effettuati hanno dimostrato che l'efficienza di questo software ha permesso di risparmiare fino al 50% dell'energia consumata precedentemente, pur mantenendo la capacità di gestire aumenti di traffico.

4.2.2 Reti Wireless

Nell'ultimo decennio si è verificata una crescita esponenziale nel consumo di dispositivi portatili come smartphone o tablet, comportando di conseguenza l'aumento proporzionale della domanda di servizi e reti mobili. Perciò i provider e gli operatori si vedono costretti sia a sostenere un ampliamento costante del volume di traffico sia a supportare allo stesso tempo diverse tipologie e generazioni di servizi. In particolare, nel settore della rete mobile si sta assistendo a un incremento della domanda di nuovi servizi che richiedono un'ampia banda di trasmissione dati come lo streaming multimediale o la comunicazione via VoIP, i quali a loro volta devono essere supportati su più tecnologie standard di rete, come ad esempio 3G oppure 4G/LTE. Tale cambiamento comporta per i fornitori dei servizi un elevato sostenimento dei costi dovuto sia all'ampliamento delle celle di accesso alla rete via radio, sia alla gestione sempre più complessa delle reti mobili.

Oltre ai difetti presenti nelle reti cablate come la scalabilità limitata o la mancanza di flessibilità di fronte all'implementazione di nuovi servizi, le reti mobili sono caratterizzate da una forte difficoltà nella gestione del traffico. Infatti i provider fanno affidamento a sistemi per il supporto alle operazioni, che richiedono competenze e risorse elevate oltre che un numero considerevole di interventi manuali, comportando di conseguenza forti rallentamenti nella gestione.



[34] B. Heller: *ElasticTree: Saving Energy in Data Center Networks*, Stanford University, 2010, pp. 1-2
[35] B. Heller: *ElasticTree: Saving Energy in Data Center Network*, op. cit.

Dal 2013, i ricercatori della Open Networking Foundation offrono una soluzione ai bisogni e le difficoltà appena descritti grazie all'insieme di vantaggi originati dall'adozione di tecnologie basate su SDN e OpenFlow. In un contesto con più reti appartenenti a diversi fornitori, invece di gestire dei gruppi distinti di dispositivi, gli operatori hanno la possibilità di coordinare e configurare rapidamente tali impianti multipli tramite strumenti basati sul controllo centralizzato. In caso di congestione nella rete, gli operatori possono applicare gli algoritmi di gestione del traffico necessari da un unico punto centrale, senza dover configurare manualmente ogni singolo dispositivo di commutazione.

Riguardo la difficoltà nell'inserimento di nuove funzioni o protocolli, l'astrazione dei dispositivi della rete permette di innovare in maniera reattiva di fronte ai mutamenti rapidi delle tecnologie mobili. Gli operatori sono in grado perciò di riprogrammare le reti in tempo reale per soddisfare ogni richiesta e necessità, riducendo drasticamente i tempi di procedimento. Infine, la capacità di applicare regole e politiche a livello atomico comporta un beneficio concreto per le reti a onde radio, permettendo di agire sulle singole sessioni o utenti connessi da diverse zone geografiche e mantenendo comunque una gestione reattiva e flessibile del traffico.[36]

4.2.3 Ricerca e sperimentazione

Come affermato in precedenza, l'attuale architettura di rete è caratterizzata da una forte rigidità che rende arduo ogni tentativo di introdurre nuovi servizi e protocolli. Di conseguenza questo aspetto negativo si presenta come una vera e propria barriera all'entrata per quanto riguarda la ricerca e la sperimentazione, provocando lo soppressione sul nascere della maggior parte delle idee innovative, senza che queste possano essere mai state testate.

L'adozione ad una tecnologia di rete come SDN permetterebbe di programmare ogni nodo destinato alla commutazione di dati in modo da isolare l'elaborazione dei flussi di traffico riservati a una o più sperimentazioni senza compromettere l'accessibilità al resto dell'utenza. Ad esempio, un ricercatore intenzionato a provare un nuovo protocollo da lui ideato su una rete accademica ha la possibilità di manipolare a suo piacimento ogni voce delle tabelle di flusso accordata secondo determinati permessi amministrativi, ciò nonostante ogni pacchetto al di fuori di tale sperimentazione, cioè appartenente ad altri utenti, sarà gestito mediante le normali procedure di inoltro e instradamento, comprendenti tutti i protocolli standard.

Infine, la rete programmabile permette di eseguire test e ricerche in maniera uniforme nonostante la presenza di router e switch appartenenti a differenti costruttori, senza dover implementare software specifico per ognuno di questi dispositivi.[37]

[36] Per un approfondimento si veda C. Kolias: *OpenFlow-Enabled Mobile and Wireless Networks*, ONF Solution Brief, 2013
[37] Per un approfondimento si veda N. McKeown: *OpenFlow: Enabling Innovation in Campus Networks*, ACM SIGCOMM Computer Communication Review, 2008

4.3 Switch nativi OpenFlow

Esistono in commercio molteplici dispositivi di commutazione compatibili con le tecnologie basate su Software-Defined Networking e di conseguenza OpenFlow. Di seguito saranno elencati alcuni dei modelli fabbricati dai maggiori produttori del settore informatico.

MODELLO		FUNZIONI
	**Cisco** *6500-6800*	• Porte Ethernet ad alta densità compatibili con MPLS e servizi VLAN; • Strumento "onePK"per gestire le operazioni su reti SDN; • Supporta il "virtual switching" (*VSS*) di ultima generazione;
	**Dell** *S4810* *S6000*	• Due sistemi operativi di serie: "Cumulus Linux" e "Big Switch Networks - Switch Light", entrambi compatibili con SDN/OpenFlow; • Possibilità di utilizzare il sistema operativo proprietario o installarne uno da zero grazie allo strumento "Open Network Install Environment";
	**HP** *5400-8200*	• Qualità di servizio (*QoS*) avanzata, basata su una classificazione del traffico secondo i livelli di rete 2, 3, e 4; • Strumenti avanzati di monitoraggio "RMON", "XRMON" e "sFlow";
	**Juniper** *Serie MX*	• Software development kit disponibile per l'integrazione e la personalizzazione dei servizi SDN; • Scalabilità 3D in termini di larghezza di banda, abbonati e servizi;
	**NEC** *PF 5240*	• Supporta OpenFlow fino alla versione 1.3.1 e ogni protocollo correlato; • Tecnologia "Virtual Tenant Network" (*VTN*) per ottimizzare l'esperienza del cloud networking;
	**Pronto** *3290*	• Possibilità di scelta tra il sistema operativo proprietario "PicOS" per reti tradizionali, oppure "Open vSwitch" compatibile con OpenFlow 1.4; • Compatibile con controller OF come "NOX", "Floodlight" e "Trema";

5. CONCLUSIONI

5.1 Benefici dell'adozione tecnologica

La scelta e quindi l'impiego di una struttura basata sul paradigma Software-Defined Networking compatibile con il protocollo OpenFlow rende la propria rete, in termini economici, un vantaggio competitivo e non un costo fisso necessario. La tecnologia SDN/OpenFlow offre un sostanziale miglioramento delle performance di banda, un adattamento dinamico alla configurazione delle applicazioni e ai bisogni commerciali, il tutto riducendo la complessità delle operazioni di manutenzione.

Tra i benefici ragguardevoli derivanti dall'adozione tecnologica di una rete programmabile sono inclusi:

- **Elevata elasticità di fronte all'innovazione**: grazie all'astrazione dell'architettura di una rete dai singoli impianti fisici come gli switch, è possibile riprogrammare quest'ultima in tempo reale secondo le necessità dell'utenza. Tutte le operazioni necessarie a personalizzare il comportamento della rete introducendo nuove caratteristiche richiedono poche ore di lavoro;

- **Minore complessità**: l'interfaccia programmabile permette di gestire la rete in maniera automatica grazie alle applicazioni che svolgono determinate mansioni oggigiorno effettuate manualmente. Questi strumenti contribuiscono alla minimizzazione dei tempi di overhead e all'aumento della stabilità di rete, due punti critici per le imprese orientate verso il "Cloud Computing";

- **Controllo atomico della rete**: il controllo centralizzato e disaccoppiato dai singoli dispositivi fornisce all'operatore una manovra precisa e granulare del traffico, come ad esempio la possibilità di applicare determinate regole per elementi specifici connessi alla rete. Ciò permette l'utilizzo di applicazioni "multi-tenant", cioè in grado di creare partizioni di istanze virtuali per ogni utente, pur mantenendo allo stesso tempo livelli elevati di sicurezza, controllo della congestione e gestione dinamica delle risorse;

- **Aumento di affidabilità e sicurezza**: uno dei vantaggi più evidenti di un'infrastruttura SDN/OpenFlow è senz'altro l'assenza di un intervento fisico su ogni dispositivo connesso ad essa ogni volta che avviene un cambiamento nella struttura o nelle politiche di inoltro, con la conseguente riduzione di guasti e errori di coerenza di tali politiche. Inoltre, la presenza di una vista panoramica della morfologia e di un controllo globale dell'infrastruttura garantisce il rispetto costante di tutte le regole e le politiche stabilite riguardanti il controllo degli accessi, qualità del servizio e sicurezza. Perciò un individuo o una azienda che adotta questa tecnologia potrà beneficiare di una maggiore liberta a livello di configurazione, di un decremento di errori di sistema e di un'osservazione più rigida delle politiche;

- **Maggiore benessere**: un'interfaccia utente a livello di applicazione che consente di prendere le redini dell'intera infrastruttura di rete conduce alla soddisfazione delle necessità dei diversi consumatori. Ad esempio, il gestore di una piattaforma online di video in streaming può decidere di modificare la qualità dell'immagine a seconda dell'abbonamento del cliente, il tutto in maniera impercettibile. Mentre i canali di video online odierni impongono all'utente la scelta della risoluzione

indipendentemente dalla capacità disponibile della rete di questi, con la tecnologia SDN/OpenFlow è possibile lasciare la decisione all'applicazione di riproduzione in grado di selezionare la qualità adatta alla banda disponibile.[38]

[38] Per un approfondimento si veda: ONF White Paper: Software-Defined Networking: The New Norm for Networks, Open Networking Foundation, 2012, pp.10-12

5.2 Atteggiamento collettivo verso SDN e OpenFlow

Nonostante i vantaggi generati dalla sostituzione di una rete tradizionale con una rete basata sul paradigma SDN, i diversi protagonisti del settore informatico, come ad esempio operatori di rete, produttori di hardware, sviluppatori di software o istituti di ricerca hanno manifestato opinioni discordanti, in parte influenzate dalla loro posizione.

Nel capitolo precedente si è discusso ampiamente sull'utilità che una rete SDN/OpenFlow può portare in campo accademico. Le università, ma anche gli istituti di ricerca percepiscono la rete programmabile come un'opportunità per abbattere le barriere dei dispositivi contenenti software proprietario in modo da minimizzare le difficoltà della ricerca, soprattutto in fase di sperimentazione. Attualmente molti campus universitari stanno installando la strumentazione necessaria per poter supportare la tecnologia SDN e OpenFlow.

Anche per quanto riguarda il "Cloud Computing" si è discusso sull'evoluzione tecnologica che otterrebbero gli impianti destinati a tali mansioni. Infatti i fornitori di servizi "Cloud" vedono l'SDN come una possibilità per sfruttare meglio i propri Data Center, grazie all'aumento di flessibilità e di semplicità nella gestione alle loro infrastrutture.

Discorso analogo per le aziende in fase di sviluppo come Nicira e BigSwitch, le quali identificano SDN e in particolare OpenFlow come un mezzo per progettare soluzioni radicali legate al mercato dell'IT, in altre parole concetti pensati ed elaborati con un metodo totalmente innovativo e assolutamente privo di vincoli rispetto alle tecnologie standard.

I principali produttori di oggi, attivi nel mercato dei dispositivi hardware di rete come ad esempio Cisco, Dell, Juniper e HP rilasciano per la maggior parte strumenti dotati di tecnologie proprietarie e spesso incompatibili con quelli di altri costruttori. Per questa ragione hanno interpretato l'avvento della tecnologia SDN come una possibile minaccia di rovina per le vendite dei loro beni. In effetti l'idea che sta alla base della rete programmabile implica la trasformazione degli attuali dispositivi di commutazione come switch o router in semplice hardware a basso costo, rendendo ogni prodotto sul mercato ugualmente appetibile agli occhi del consumatore indipendentemente dal marchio di fabbrica. In risposta a questo sintomo di fallimento, la maggior parte di queste imprese, come già mostrato, ha prodotto una propria variante di dispositivo basato su SDN/OpenFlow, che in alcuni casi però è dotato di una programmabilità limitata dovuta dall'imposizione di un sistema operativo proprietario. Tuttavia alcuni di questi produttori si stanno orientando verso lo sviluppo di apparecchi provvisti di un'interfaccia interamente aperta e stanno acquisendo imprese giovani con progetti e idee rivolte verso la tecnologia SDN affinché rimangano competitivi nel settore e raggiungano una forte elasticità di fronte alle variazioni della domanda di mercato.

Le maggiori aziende nate sul web come Google, Facebook, Amazon o Yahoo non solo fanno già uso di reti compatibili con SDN/OpenFlow, in particolare per gestire i propri Data Center, ma sono tra i principali fondatori e ricercatori dell'organizzazione più attiva nello sviluppo e nella promozione della tecnologia di rete programmabile: la Open Networking Foundation.[39]

Gli sviluppatori di software e i produttori di hardware sono entrambi entusiasti davanti alla probabile diffusione della tecnologia SDN: i primi hanno la possibilità di ampliare in maniera smisurata la realizzazione di nuove applicazioni destinate ai controller ed eterogenee a seconda delle funzioni richieste,

[39] Per un approfondimento si veda: *https://www.opennetworking.org/about/board-of-directors*

come ad esempio il traffic engeenering per un Data Center oppure la semplice gestione di una rete aziendale; i secondi, in contrasto con i produttori di switch, hanno l'opportunità di incrementare il loro mercato grazie alla richiesta di strumentazione hardware generica senza alcune caratteristiche applicative di alto livello.

Infine gli operatori della rete non si sono ancora interamente esposti in relazione al fenomeno della rete programmabile. Anche se, come già discusso precedentemente, l'adozione di questa tecnologia porterebbe grandi benefici ai service provider, in particolare per quanto riguarda la distribuzione dei servizi telematici per ogni utente, la maggior parte di queste aziende è in attesa di comprendere quale variante di SDN può diventare lo standard dominante del mercato: quella attualmente proposta da determinati venditori di strumenti di rete, cioè comprensiva di un'API proprietaria e chiusa, oppure la versione più pura del Software-Defined Networking, ovvero completamente indipendente dal marchio di produzione e dotata di un'interfaccia aperta e gestita interamente dal software del controller.

5.3 Previsioni Future

Nei capitoli precedenti si è fatta luce su quanto la tecnologia programmabile SDN e in particolare OpenFlow siano promettenti nel campo delle reti. Come si è visto, sono state sviluppate molteplici applicazioni per ogni area del settore informatico, come gestione del traffico, sicurezza, comunicazione senza fili, o viabilità. Tuttavia attualmente sono state realizzate reti basate su SDN/OpenFlow a dimensione limitata, perciò la sfida maggiore per questa tecnologia comprende senz'altro il suo utilizzo nelle reti geografiche o "Wide Area Network" (*WAN*). Finora sono state effettuate solamente delle simulazioni di reti WAN basate su OpenFlow che hanno rafforzato l'eventualità di un controllo esteso, ma tuttavia non hanno ancora estinto alcuni dubbi riguardanti la scalabilità elevata e i livelli di performance.[40]

Nonostante sia già presente una notevole produzione di dispositivi e software compatibile con tecnologie SDN/Openflow, esiste uno sforzo altrettanto considerevole per quanto riguarda la ricerca e progettazione di meccanismi provvisti di nuove funzionalità. In particolare sono state proposte diverse idee relative alla gestione delle tabelle di flusso, come la distinzione di voci con vita lunga o breve per limitare tempi di reattività degli switch e i tempi di overhead per i controller. È stato inoltre dimostrato che un controller può singolarmente gestire fino a 6 milioni di richieste di flusso, ciò nonostante, per aumentarne l'efficienza, la scalabilità e soprattutto l'affidabilità, si è considerata l'idea di un controllo centralizzato logicamente, ma distribuito fisicamente nella rete.

Un altra lacuna già affrontata precedentemente concerne l'interazione tra controller e interfaccia utente, in quanto oggigiorno non esiste uno standard della "Northbound Interface". La ragione più plausibile di questa mancanza è spiegabile dal fatto che questa interazione, a differenza della "Southbound Interface", non implica un'implementazione a livello di hardware. Diverse organizzazioni implicate nello sviluppo dei controller hanno proposto l'utilizzo di un linguaggio di configurazione delle reti in grado di esprimere le regole e politiche di inoltro affinché esse siano recepite e convertite correttamente dalle applicazioni di alto livello. Finché non emergerà uno standard dominante per questa interfaccia si dovrà fare affidamento a soluzioni specifiche e dipendenti dal tipo di infrastruttura da gestire.

Infine, la domanda crescente di traffico mobile sta dando un'impronta eterogenea alla navigazione in rete. I vari utenti si connettono mediante differenti metodi quali wireless locale, mobile, o il tradizionale cablaggio indipendentemente dalla zona di accesso. Inoltre, l'efficienza nella consegna dei contenuti e nella gestione delle risorse sta diventando fondamentale soprattutto per l'accesso alle reti mobili. La rete programmabile secondo il paradigma SDN/OpenFlow ha il potenziale per semplificare lo sviluppo e la gestione di tali servizi e applicazioni con totale efficienza. Nonostante si possa pensare che il concetto di controllo centralizzato entri in conflitto con il livello strutturale distribuito delle reti mobili, grazie a progetti come ad esempio OpenRoads, già citato nei capitoli precedenti, si è dimostrato che anche in questo ambiente è possibile ottenere risultati impressionanti concedendo comunque massima libertà agli utenti interconnessi.[41]

[40] Per un approfondimento si veda: A. Lara: *Network Innovation using OpenFlow: A Survey*, IEEE Communications Surveys & Tutorials, 2013, pp. 16-17

[41] Per approfondimento si veda: M. Mendonca: *A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks*, IEEE Communications Surveys & Tutorials, 2013, pp.11-13

BIBLIOGRAFIA

[1] M. Bansal, J. Mehlman, S. Katti, P. Levis: *OpenRadio: A Programmable Wireless Dataplane*, Stanford University, 2012.

[2] Z. Cai A. Cox T. Ng: *Maestro: A System for Scalable OpenFlow Control*, Rice University, 2010.

[3] M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeown, S. Shenker: *Ethane: Taking Control of the Enterprise*, Sigcomm, 2007.

[4] S. Das, A. R. Sharafat: *MPLS with a Simple OPEN Control Plane*, Stanford University, 2011.

[5] A. D. Ferguson, A. Guha, C. Liang, R. Fonseca, S. Krishnamurthi: *Hierarchical Policies for Software Defined Networks*, Brown University, 2012.

[6] N. Gude, T. Koponen, J. Pettit, B. Pfa, M. Casado, N. McKeown, S. Shenker: *Nox: towards an operating system for networks*, ACM SIGCOMM Computer Communication Review, 2008.

[7] N. Handigol, S. Seetharaman, M. Flajslik, N. McKeown, R. Johari: *Plug-n-Serve: Load-Balancing Web Traffic using OpenFlow*, Stanford University, 2009

[8] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, N. McKeown: *ElasticTree: Saving Energy in Data Center Networks*, Stanford University, 2010.

[9] J. H. Jafarian, E. Al-Shaer, Q. Duan: *OpenFlow Random Host Mutation: Transparent Moving Target Defense using Software Defined Networking*, University of North Carolina, 2012.

[10] A. Khurshid, X. Zou, W. Zhou, M. Caesar, P. Godfrey: *VeriFlow: Verifying Network-Wide Invariants in Real Time*, University of Illinois, 2012.

[11] C. Kolias: *OpenFlow-Enabled Mobile and Wireless Networks*, Open Networking Foundation, 2013.

[12] J. F. Kurose, K. Ross: *Reti di Calcolatori e Internet*, Pearson 2007.

[13] A. Lara, A. Kolasani, B. Ramamurthy: *Network Innovation using OpenFlow: A Survey*, IEEE Communications Surveys & Tutorials, 2013.

[14] D. M. F. Mattos, N. C. Fernandes, V. T. da Costa, L. P. Cardoso, M. E. M. Campista, L. H. M. K. Costa, O. C. M. B. Duarte: *OMNI: OpenFlow MaNagement Infrastructure*, Universidade Federal do Rio de Janeiro, 2011.

[15] M. McBride: SDN Security Considerations in the Data Center, Open Networking Foundation, 2013.

[16] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. peterson, J. Rexford, S. Shenker, J. Turner: *OpenFlow: Enabling Innovation in Campus Networks*, ACM SIGCOMM Computer Communication Review, 2008.

[17] M. Mendonca, B. Astuto, A. Nunes, X. Nguyen, K. Obraczka, T. Turletti: *A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks*, IEEE Communications Surveys & Tutorials, 2013.

[18] Open Networking Foundation: *OpenFlow Switch Specification*, ONF, 2014.

[19] Open Networking Foundation: *SDN Architecture Overview*, ONF White Paper, 2013.

[20] Open Networking Foundation: *Software-Defined Networking: The New Norm for Networks*, ONF White Paper, 2012.

[21] S. Raza, D. Lenrow: *Northbound Interface*, Open Networking Foundation, 2013.

[22] J. Suh, H. Choi, W. Yoon, T. You, T. Kwon, Y. Choi: *Implementation of Content-oriented Networking Architecture (CONA): A Focus on DDoS Countermeasure, Seoul National University*, 2010

[23] L. Suresh, J. Schulz-Zander, R. Merz, A. Feldmann, T. Vazao: *Towards Programmable Enterprise WLANs with Odin*, Proceedings of the first workshop on Hot topics in software defined networks, 2012.

[24] A. Tootoonchian, S. Gorbunov, Y. Ganjali: *On Controller Performance in Software-Defined Networks*, USENIX conference, 2012.

[25] A. Voellmy, J. Wang: Scalable Software Defined Network Controllers, SIGCOMM, 2012.

[26] Y. Yamasaki: *Flexible Access Management System for Campus VLAN Based on OpenFlow*, IEEE/IPSJ 11th International Symposium on Applications and the Internet, 2011.

[27] G. Yao, J. Bi, P. Xiao: *Source Address Validation Solution with OpenFlow/NOX Architecture*, Tsinghua University, 2011.

[28] K. Yap, M. Kobayashi, R. Sherwood, N. Handigol, T. Huang, M. Chan, N. McKeown: *OpenRoads: Empowering Research in Mobile Networks*, Stanford University, 2009.

SITOGRAFIA

[1] http://archive.openflow.org/wk/index.php/MPLS_with_OpenFlow/SDN/

[2] http://www.kandoo.org/

[3] http://www.noxrepo.org/

[4] https://openflow.stanford.edu/display/Beacon/Home/

[5] https://www.opennetworking.org/about/board-of-directors/

[6] http://www.projectfloodlight.org/floodlight/

[7] https://www.sdncentral.com/projects/trema/