Università di Pisa

Corso di Laurea Magistrale in Ingegneria delle Telecomunicazioni
Dipartimento di Ingegneria dell'Informazione

Progetto, realizzazione e validazione sperimentale di un'applicazione SDN per Traffic Recovery e Load Balancing

Candidato:
Nicola Santinelli

Relatori:
Prof. Stefano Giordano
Prof. Michele Pagano
Ing. Davide Adami

Sessione di Laurea del 28/04/2014
Anno Accademico 2012/2013

Riassunto analitico

Nel corso degli ultimi anni, la comunità scientifica si è prodigata nell'esplorazione delle potenzialità fornite dal Software-Defined Networking, un paradigma che prevede la netta separazione tra piano dati e piano di controllo all'interno di una rete. Così facendo, il comportamento dei dispositivi risulta programmabile tramite un'applicazione di controllo.

In questo contesto, il presente elaborato documenta il lavoro di tesi intrapreso per progettare, realizzare e validare una tale applicazione avente funzionalità di Traffic Recovery e Load Balancing. Le stime riguardanti l'utilizzazione dei collegamenti e la banda occupata dai vari flussi, così come la rivelazione dei malfunzionamenti, sono state gestite sfruttando le peculiarità del protocollo OpenFlow.

L'aspetto relativo alla Traffic Recovery è stato affrontato con strategie sia reattive che proattive. In particolare, il traffico generato dalle sorgenti può essere suddiviso in quattro classi, caratterizzate da un livello di protezione via via crescente. Riguardo alla funzionalità di Load Balancing, sono state adottate e confrontate tre possibili metriche per l'assegnazione dei costi ai vari collegamenti, in funzione del carico da essi sperimentato.

Parole chiave: Software-Defined Networking, OpenFlow, Traffic Recovery, Protection, Restoration, Load Balancing.

Abstract

During the last years, Software-Defined Networking (SDN) has been widely explored in order to understand the possibilities provided by this kind of architecture. Introducing a clean separation between the control plane and the data plane within the network, SDN allows a control application to program the behavior of the underlying infrastructure.

This thesis deals with designing, fulfilling and validating such an application with Traffic Recovery and Load Balancing tasks. The features of the OpenFlow protocol have been exploited in order to provide an estimate of links utilization and flows bandwidth, as well as to effectively detect link failures.

Traffic Recovery has been handled adopting both reactive and proactive strategies. More precisely, network traffic can be grouped into four classes with an increasing level of protection. With respect to Load Balancing, three alternative ways of assigning a cost to a certain link, as a function of its load, have been defined and compared.

Keywords: Software-Defined Networking, OpenFlow, Traffic Recovery, Protection, Restoration, Load Balancing.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

Nel corso degli ultimi anni, la comunità scientifica si è prodigata nell'esplorazione delle potenzialità fornite dal *Software-Defined Networking* (SDN), un paradigma che prevede la netta separazione tra piano dati e piano di controllo all'interno di una rete (paragrafo 2.2). Così facendo, il comportamento dei dispositivi risulta non soltanto configurabile ma anche *programmabile* da remoto, tramite un'applicazione di controllo.

In questo contesto, il presente elaborato documenta il lavoro di tesi intrapreso per progettare, realizzare e validare una tale applicazione avente funzionalità di *Load Balancing* e *Traffic Recovery*.

Load Balancing Il fine ultimo di una strategia di tipo Load Balancing (LB) è, letteralmente, quello di ottenere un "bilanciamento del carico" sperimentato all'interno dell'infrastruttura. Da un punto di vista pratico, ciò che si tenta di fare è distribuire equamente il traffico generato dagli host della rete su, potenzialmente, tutti i collegamenti (*links*) di cui si dispone. Il problema consiste dunque nel determinare, di volta in volta, il percorso[1] *migliore* sul quale inoltrare i pacchetti appartenenti ad un certo flusso[2].

Nel mondo delle reti tradizionali, tale problematica è nota con il termine di *routing* e la sua risoluzione è affidata a protocolli distribuiti, denominati *Interior Gateway Protocols* (IGP). Tra i più popolari IGP attualmente in circolazione si ricordano RIP, OSPF ed IS-IS. A meno di configurazioni particolari, detti protocolli intendono la bontà di un percorso come una proprietà *nota a priori*, specificata cioè una volta per tutte dall'amministratore della rete oppure determinata autonomamente dall'IGP considerato, del tutto slegata dalle particolari

[1] Un "percorso" (*path*) è caratterizzato da un router (switch) d'ingresso, da un router (switch) di uscita e da un insieme di router (switch) e link intermedi, sui quali vengono inoltrati i pacchetti appartenenti ad un flusso.

[2] Con il termine "flusso" (*flow*) s'intende una successione di pacchetti accomunati dai valori assunti da un insieme di campi presenti nelle intestazioni dei pacchetti stessi, delle trame che li contengono o dei segmenti in essi contenuti.

condizioni nelle quali versa l'infrastruttura. Tale approccio, per quanto semplice ed intuitivo, potrebbe risultare non appropriato.

Si supponga, ad esempio, di disporre di due sottoreti separate da un "dominio di routing", ovvero da un insieme di router (se si tratta di una rete tradizionale) o di switch (nel caso di rete SDN) arbitrariamente connessi[3] (paragrafo 6.1). In uno scenario di questo tipo, la pratica di cui sopra porterebbe all'utilizzazione di *due soli* percorsi tra tutti quelli possibili, peraltro costituiti dallo stesso insieme di collegamenti da attraversare in entrambe le direzioni[4]. Chiaramente, non è possibile parlare di LB.

Un'eventualità come quella appena descritta risulta sconveniente, quantomeno da due punti di vista. In prima battuta, un operatore che abbia investito consistenti somme di denaro per mettere in campo la propria infrastruttura non sarà sicuramente disposto a vederla, per larga parte, inutilizzata. Secondariamente, anche ammesso di poter trascurare gli aspetti economici riguardanti la gestione della propria rete, permane un problema dal punto di vista delle prestazioni. Al crescere del numero di flussi simultaneamente attivi, infatti, aumenta il livello di congestione dei pochi link servibili. A causa della mancanza di "strade" alternative, grazie alle quali sarebbe possibile alleggerire il carico da essi sperimentato, ciò si traduce inevitabilmente in:

- un incremento delle percentuali di perdita;
- una riduzione della banda[5] ottenibile da ciascun flusso.

Per avvicinarsi maggiormente ad una spartizione equa del traffico generato all'interno della rete, si potrebbe pensare d'intendere la bontà di un percorso come una proprietà variabile nel tempo, dipendente cioè dall'attuale livello di *congestione* sperimentato dai link che lo costituiscono. Nel capitolo 5 verrà illustrato come l'applicazione implementata sia in grado di raggiungere un tale obiettivo. Per adesso, ci preme solamente enfatizzare i due aspetti cardine della strategia impiegata:

1. stima in tempo reale dell'utilizzazione di ogni collegamento all'interno della rete;
2. definizione di una metrica che mappi il livello di utilizzazione stimato in un nuovo valore, avente il significato di *costo* del collegamento.

Il primo punto è stato gestito sfruttando le peculiarità del protocollo OpenFlow, il quale costituisce la più diffusa tra le interfacce utilizzabili dal piano di controllo per comunicare con i dispositivi di rete. Riguardo al secondo, sono state adottate e confrontate tre metriche alternative per l'assegnazione dei costi ai vari collegamenti, in funzione del carico da essi sperimentato.

[3] In realtà, si richiede che per ogni coppia di router (switch) della rete esista almeno un percorso in grado di connetterli.

[4] Stiamo supponendo che i link siano bidirezionali.

[5] Una precisazione è d'obbligo: nel seguito, il termine "banda" sarà sempre inteso come *rate*, adottando la prassi valida nel mondo delle reti di telecomunicazioni. Dimensionalmente, dunque, la banda verrà espressa in 'bit/s' (o in unità di misura equivalenti, quali 'byte/s' e 'pacchetti/s').

Traffic Recovery La terminologia utilizzata in questo e nei restanti capitoli, relativamente a tali problematiche, trae ispirazione da quella impiegata in [39].

La funzionalità di Traffic Recovery (TR) consiste essenzialmente nella capacità di reagire ad un malfunzionamento che colpisce un percorso attivo. Tale reazione si manifesta attraverso l'attivazione di un nuovo percorso, alternativo al primo. Ai fini della comprensione di ciò che verrà esposto nel seguito, è bene sottolineare come il *calcolo* di un percorso e la sua *attivazione* siano due operazioni non necessariamente concomitanti.

Come risulterà chiaro dalla lettura del presente elaborato, in condizioni di funzionamento ordinario, il primo pacchetto di un nuovo flusso verrà recapitato al piano di controllo, il quale, secondo un proprio algoritmo (paragrafo 5.5), deciderà quale percorso debbano seguire i pacchetti corrispondenti e provvederà ad istruire coerentemente gli switch interessati. Un percorso di questo tipo viene indicato come "percorso working" (*Working Path*, WP) per il flusso in esame. Supponiamo che un guasto colpisca uno (o più di uno) dei link costituenti il WP, ovvero che quest'ultimo risulti inutilizzabile. Per mantenere integra la comunicazione ent-to-end in tali circostanze, si rende necessario inoltrare il traffico su un nuovo percorso, chiamato anche "percorso recovery" (*Recovery Path*, RP) per il flusso in esame. Riassumendo: un WP trasporta traffico utente in condizioni "normali"; un RP trasporta traffico utente quando il relativo WP si guasta. È importante precisare come uno stesso percorso possa risultare "working" per un certo flusso e, allo stesso tempo, "recovery" per un altro. Quest'ultimo aspetto differenzia le tecniche di TR adottate in questo lavoro da quelle tipicamente impiegate nelle reti ottiche di trasporto.

Il concetto di TR può essere scisso ulteriormente in *Protection* e *Restoration*. La differenza tra le due sfaccettature è legata al tipo di allocazione delle risorse portata avanti durante l'attivazione del RP. Nel caso in cui venga adottato un approccio di tipo Protection, il calcolo del RP viene effettuato contestualmente al calcolo del WP. Si tratta cioè di una strategia "proattiva" (*proactive*) o "preventiva". Si osservi come, in questo caso, WP e RP debbano risultare "disgiunti nei link" (*link disjoint*), in modo da assicurare che un singolo guasto non vada a compromettere il funzionamento di entrambi. Qualora, invece, la scelta ricadesse su un approccio di tipo Restoration, il calcolo del RP avverrebbe solamente in corrispondenza di un malfunzionamento interessante il relativo WP. Siamo cioè di fronte ad una strategia "reattiva" (*reactive*).

In questo lavoro di tesi, abbiamo scelto di consentire la suddivisione del traffico generato dalle sorgenti in quattro classi, caratterizzate da un livello di protezione via via crescente. Per i flussi appartenenti alla classe "meno prioritaria", la tecnica di TR adottata è quella della Restoration. Il traffico generato dalle successive, invece, gode di una strategia di tipo Protection. Per quei flussi definiti dall'operatore come particolarmente "sensibili", infine, è prevista una variante di Protection ancora più robusta di quella già menzionata. I pacchetti appartenenti a detti flussi, infatti, vengono inoltrati *contemporaneamente* su due percorsi disgiunti, ciascuno dei quali può essere inteso come WP.

Struttura dell'elaborato Il resto del documento è organizzato come segue. Nel capitolo 2 viene fornita la definizione formale di Software-Defined Networking e la descrizione della relativa architettura stratificata. In aggiunta, sono citate le motivazioni che ne hanno spinto l'adozione da parte della comunità scientifica, i principali benefici derivanti dal suo utilizzo ed una sintesi delle iniziative intraprese da alcune tra le organizzazioni più attive in questo ambito.

Il capitolo 3 tratta approfonditamente del protocollo OpenFlow ed in particolare della specifica 1.0.0. Vengono inoltre passate in rassegna alcune problematiche caratterizzanti le architetture SDN che sfruttano tale protocollo.

Nel capitolo 4 sono raccolti alcuni cenni sull'emulatore Mininet e sulla piattaforma di controllo POX. Grazie al primo è stato possibile generare le topologie di rete considerate, mentre le API[6] del secondo hanno consentito la scrittura dell'applicazione di controllo, oggetto di questo lavoro di tesi. Detta applicazione viene esaminata nel dettaglio all'interno del capitolo 5 ed il codice corrispondente può essere recuperato consultando l'appendice D.

Il capitolo 6 passa in rassegna le topologie di rete impiegate all'interno degli esperimenti condotti sull'architettura implementata. La descrizione di detti esperimenti e la raccolta dei risultati corrispondenti, espressi principalmente sotto forma di grafici, sono affidate congiuntamente al capitolo 7 ed alle appendici A, B e C.

Infine, il capitolo 8 conclude l'elaborato.

[6] Un'*Application Programming Interface* (API) specifica come alcuni componenti software debbano interagire gli uni con gli altri. In pratica, un'API viene frequentemente espressa sotto forma di libreria. Al suo interno si possono trovare le specifiche di metodi, strutture dati, classi e variabili.

Capitolo 8

Conclusioni

Questo elaborato documenta il lavoro di tesi intrapreso per progettare, realizzare e validare un'applicazione di controllo, conforme al Software-Defined Networking, avente funzionalità di Load Balancing e Traffic Recovery.

Il Load Balancing viene ottenuto sfruttando le statistiche relative alle porte di uno switch che il protocollo OpenFlow mette a disposizione. A patto di giungere ad una rappresentazione astratta dell'infrastruttura sotto forma di grafo, tramite tali informazioni è possibile far variare il costo di un certo arco in funzione del livello di utilizzazione sperimentato dal collegamento associato. La conseguente esecuzione dell'algoritmo di Dijkstra consente di determinare i cammini corrispondenti ai percorsi attualmente meno congestionati, sui quali inoltrare il traffico della rete. L'applicazione prevede, inoltre, la possibilità di utilizzare tre diverse tipologie di metrica, le quali si differenziano per la velocità con cui un certo link viene penalizzato.

Gli esperimenti condotti dimostrano che, grazie alla nostra euristica di Load Balancing, un flusso UDP sperimenta in media percentuali di perdita inferiori, rispetto a quelle che subirebbe qualora si dovesse adottare una strategia statica di assegnazione dei costi. Allo stesso tempo, un flusso TCP è in grado di ottenere rate trasmissivi mediamente superiori, se confrontati con quelli ottenibili optando per un criterio di tipo Fixed Cost.

La funzionalità di Traffic Recovery viene, a sua volta, implementata sfruttando le peculiarità del protocollo OpenFlow. Grazie al messaggio asincrono Port-Status, siamo in grado di rivelare tempestivamente l'esistenza di un malfunzionamento. Nel ricollocare i flussi interessati, si procede in senso decrescente rispetto alla banda da essi occupata. Quest'ultima viene calcolata grazie alle statistiche di flusso prelevabili dalla tabella di uno switch.

L'applicazione è in grado di gestire fino a quattro classi di traffico, le quali si differenziano per il livello di protezione riservato ai flussi corrispondenti. La tecnica di Traffic Recovery associata ad un flusso Bronze è quella della Restoration. Ciò significa che un eventuale Recovery Path viene calcolato solamente a guasto avvenuto. Esso può coincidere sia con il percorso meno congestionato che con quello maggiormente congestionato disponibile al momento, ma pur

sempre dotato di banda residua sufficiente ad accogliere il nuovo flusso. La tecnica di recovery riservata alle classi di traffico successive, invece, è quella della Protection. Il Recovery Path per tali flussi viene infatti calcolato contestualmente all'installazione del relativo Working Path. Per un flusso Silver, le regole relative al primo percorso sono caratterizzate da una priorità inferiore, rispetto a quelle utilizzate per il secondo. A seguito di un guasto, dunque, il Working Path deve essere rimosso dalle tabelle degli switch perché il Recovery Path possa attivarsi. I pacchetti appartenenti ai flussi Gold e Platinum, invece, vengono inoltrati contemporaneamente su due percorsi disgiunti, risultando per questo immuni nei confronti di guasti isolati. In aggiunta, la classe Platinum gode dell'isolamento fisico rispetto al resto dell'infrastruttura.

I test dimostrano che il Time To Repair sperimentato da un flusso Bronze è in media superiore rispetto a quello sperimentato da un flusso Silver, a causa del maggior numero di operazioni richieste per attuare la funzionalità di Traffic Recovery. In particolare, l'applicazione consente di ottenere un valore di Mean Time To Repair inferiore a 40 e 30 millisecondi, rispettivamente. Quest'ultimo risultato è particolarmente significativo in quanto consente di affermare che, grazie alle strategie di recovery adottate, il vincolo dei 50 millisecondi tipico delle reti ottiche di trasporto è mediamente rispettato.

Infine, segnaliamo che il lavoro di tesi presentato ha fornito il materiale utile per la stesura dell'articolo "Design and Performance Assessment of an SDN Control Application for Load Balancing and Traffic Recovery", candidato ad essere pubblicato negli IEEE GLOBECOM 2014 Proceedings.