**Proposta soluzione proattiva self healing ring**

Obiettivi:

* Compilare Flow Table di ciascuno switch in modo differente per garantire il funzionamento voluto
* Compilare Group Table di ciascuno switch in modo differente per garantire il funzionamento voluto
* Scrivere un codice generalizzato che possa funzionare con un numero arbitrario di switch

**Step 1: Topology discovery**

Per conoscere la topologia della rete (quanti switch, quanti link, quanti host e quali porte collegano cosa) si fa girare una funzione di topology discovery che implementa **get\_switch, get\_link e get\_host\*.**

@set\_ev\_cls(event.EventSwitchEnter)  
**def** get\_topology\_data(self, ev):  
 switch\_list = get\_switch(self.topology\_api\_app, **None**)  
 switches = [switch.dp.id **for** switch **in** switch\_list]  
 links\_list = get\_link(self.topology\_api\_app, **None**)  
 links = [(link.src.dpid, link.dst.dpid, {**'port'**: link.src.port\_no}) **for** link **in** links\_list]

\***nota:** manca get\_host da implementare

Idealmente questa funzione dovrebbe buttare fuori 3 liste **switches**, **links** e **hosts** (che manca, ma la cui struttura potrebbe essere grosso modo come quella di switches).

**Idea**: si potrebbe scrivere una semplice funzione che prende le tre liste e restituisce una matrice così fatta

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id switch | Porta per senso orario | Porta per senso antiorario | Porta per host | Id host |
| ….. | …. | …….. | ….. | ….. |
| ……. | ……. | ……. | …. | ….. |

**Step 2: compilazione tabelle**

Per compilare le tables in modo differente su ciascuno switch si potrebbero implementare le funzioni **flow\_mod** e **group\_mod** (che modificano le rispettive tabelle) all’interno di un ciclo for che scorre le righe della matrice sopra. Ad ogni iterazione del ciclo si eseguono le **flow\_mod** e **group\_mod** passando i parametri della riga selezionata.

In questo modo dovremmo riuscire a compilare le tabelle in modo differente per ogni switch, per qualsiasi numero di switches.

Per quanto riguarda le specifiche tabelle, il percorso di un pacchetto dovrebbe essere

* Flow table 🡪 matching in base a porta di arrivo, indirizzo di dest e indirizzo sorgente
  + Se destinato all’host 🡪 manda all’host
  + Se broadcast madato da me o già visto 🡪 drop
  + Altrimenti manda alla group table (group id 1 se viene da dx, group id 2 se viene da sx)
* Group table 🡪 fast failover, 2 buckets per group id