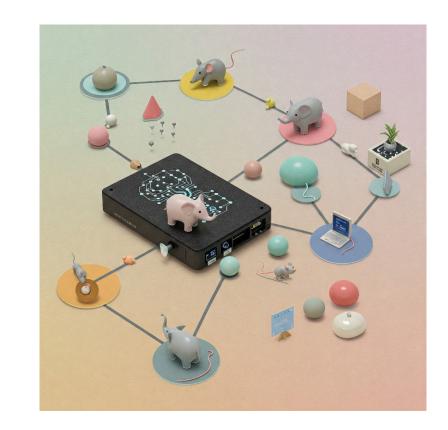
PROGETTO 2 IDENTIFICAZIONE E GESTIONE DI «ELEFANTI»

Obiettivo del progetto

- **Obiettivo** \geq : sviluppo di un controller Ryu in grado di discriminare le connessioni TCP tra «elefanti» e «topi» sulla base del volume di traffico coinvolto. I «topi» sono gestiti direttamente del controller; quando, però, la trasmissione viene promossa a «elefante», il controllore installa su tutti gli switch coinvolti regole per l'offloading del traffico al dataplane migliorando le performance.
- **Topologia** : idealmente il progetto ha senso su qualunque topologia; per la dimostrazione è stata scelta una rete lineare con 2 host e 3 switch. Si assume schema di rete statico.
- Client e Server : un server lPerf è in esecuzione su uno degli host per permettere misure di traffico da un client.



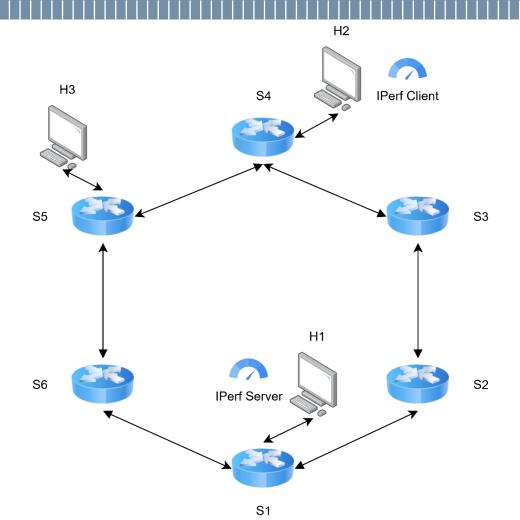
Obiettivo del progetto

- **Eventi** : il controller inizializza gli switch e impara la topologia rimanendo in attesa degli appositi trigger di Ryu. All'arrivo dei pacchetti esso agisce da ARP Proxy, se necessario, oppure esegue l'instradamento Hop-by-Hop. In caso di traffico TCP il controllore tiene traccia del volume.
- Validazione ✓: viene valutato l'RTT rispetto al totale dei byte trasmessi. Al superamento di un arbitrario limite, il controller installa le regole di forwarding sugli switch; non essendo più necessario eseguire l'inoltro di tutti i pacchetti passando per il control plane questi risultano inoltrati molto più velocemente.



Scenario di riferimento (Mininet) Modello ideale

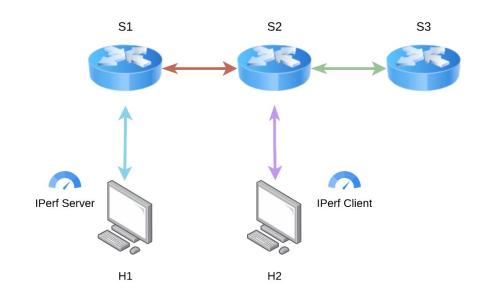
Topologia \rightleftharpoons : anello con 6 switch (S^*) e 3 host (H^*).



▲ N.B: È rappresentato solo il dataplane! Il control-plane è out-of-band.

Scenario di riferimento (<u>Testbed</u>) Modello reale

Topologia \blacksquare : lineare con 3 switch (S^*) e 2 host (H^*).



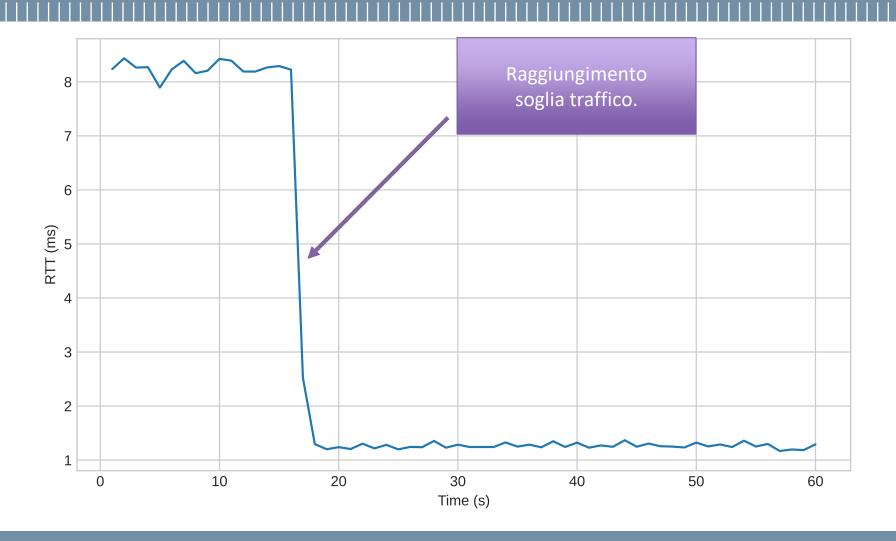
▲ N.B: sul diagramma è rappresentato solo il dataplane! (In foto, i cavi gialli.)



Funzionamento

- Ryu riceve una trama Ethernet da uno switch.
 - ➤ Se ARP Request → Esegui ARP Proxy per rispondere senza generare traffico broadcast.
 - Se pacchetto <u>IP(v4)</u> → Individua la porta corretta sui cui inoltrare il messaggio in modo da raggiungere la destinazione o il designato next-hop.
 - Se pacchetto <u>TCP</u> → Aggiungi la connessione nella lista di sessioni monitorate; aggiorna il contatore dei dati trasmessi ad essa relativa.
 - La connessione TCP ha raggiunto il threshold → Installa la regola di forwarding sullo switch che ha recapitato il pacchetto.
- Le regole si propagano «a cascata»!
- Il controller provvede periodicamente a rimuovere le connessioni inattive dalle sue strutture dati.
 Per gli «elefanti» in gestione al dataplane avviene la rimozione automatica delle regole inutilizzate sfruttando direttamente i timer di OpenFlow.

Dimostrazione



Mininet-vs-Testbed

- > ② A differenza della simulazione effettuata su Mininet, per la dimostrazione su testbed abbiamo:
 - Cambiato la metrica di riferimento: RTT vs Throughput TCP;
 - Semplificato la topologia di rete: **Toroidale (6SW, 3HS) vs Lineare (3SW, 2HS)**.
- Perché? Gli switch hanno mostrato comportamenti anomali in presenza di alto traffico sul control plane (riavvi inattesi, pacchetti non recapitati al controller).
 - Validazione con iPerf3 a bitrate vincolato, **l'RTT è una buona variabile libera**.
- Gli switch sono risultati stabili a fronte di 128 kbit/s di throughput sul piano di controllo.

This bad boy can fit 2 WHOLE VoIP sessions

