11th Israeli Networking Day

Seamless SDN Route Updates
Sylvie Delaet, Shlomi Dolev, Daniel Khankin, Shimrit Tzur-David and Tomer Godinger
Contents

• Introduction
• Software Defined Network Settings
• Multicast Based SDN Route Update
  • Sequential Replacement Algorithm
  • Control Clearance Replacement Technique
• Simulation Setup and Results
• Conclusions
Introduction
Introduction: Route Update Problem
Software Defined Network Settings
\[ s = n_0 \rightarrow n_{10} \rightarrow n_3 \rightarrow n_4 \rightarrow n_5 \rightarrow n_7 \rightarrow n_{13} \rightarrow n_{14} \rightarrow d = n_{15} \]

\[ n_2 \rightarrow n_1 \rightarrow n_6 \rightarrow n_{12} \rightarrow n_{11} \]
Software Defined Network Settings

- Constraints that we need to ensure:
  - The controller updates while current route is operational.
  - The sender does not stop sending packets
  - The receiver does not stop receiving packets
  - No packets are lost
  - Packets arrive in a FIFO order
Multicast Based SDN Route Update
\( s = c_0 \)

\( s \rightarrow d \)

\( (s, d) \rightarrow c_1 \)

\( (s, d) \rightarrow n_1 \)

\( s \rightarrow d \)

\( s \rightarrow c_1 \)

\( c_1 \rightarrow c_2 = n_6 \)

\( c_2 = n_6 \rightarrow c_3 = n_3 \)

\( c_3 = n_3 \rightarrow d = c_4 \)

\( \cdots \)
Sequential Replacement Algorithm
$C$ - old route

$s = c_0 \rightarrow c_1 \rightarrow c_2 \rightarrow c_3 \rightarrow c_4 \rightarrow c_5 \rightarrow c_6 \rightarrow c_7 \rightarrow c_8 \rightarrow c_9 \rightarrow c_{10} \rightarrow c_{11} \rightarrow c_{12} \rightarrow d = c_{13}$
$n_0 = c_0 = n_0$

$\begin{align*}
    n_1 & \rightarrow n_2 \\
    n_2 & \rightarrow n_1 \\
    n_3 & \rightarrow n_4 \\
    n_4 & \rightarrow n_3 \\
    n_5 & \rightarrow n_6 \\
    n_6 & \rightarrow n_5 \\
    n_7 & \rightarrow n_8 \\
    n_8 & \rightarrow n_7 \\
    n_9 & \rightarrow n_10 \\
    n_10 & \rightarrow n_9 \\
    n_{11} & \rightarrow n_{12} \\
    n_{12} & \rightarrow n_{11} \\
    n_{13} & \rightarrow n_{14} \\
    n_{14} & \rightarrow n_{13} \\
    d & = n_{15}
\end{align*}$
$NIC = N \cap C$
\[ [n \; \text{nic}_{\text{last}, d}]_C = [\text{nic}_{\text{last}, d}]_N \]

\[ \text{S} \]

\[ c_0 = n_0 \quad c_2 = n_{10} \quad c_5 = n_3 \quad c_6 = n_4 \quad c_7 = n_5 \quad c_9 = n_7 \quad c_{10} = n_{13} \quad c_{11} = n_{14} \quad c_{13} = n_{15} \]

\[ d \]

\[ \text{nici}_{y+1} \]

[Diagram of network with nodes labeled and edges connecting them, indicating the path from S to d through various intermediate nodes labeled with n values.]
$n_{ic_y}$
\[ n_{ic_y} \]

\[ n_{ic_{y+1}} \]
SR Algorithm Summary

• None of the routers need to buffer any of the packets
  • Packets arrive in order
  • Copy of $p_2$ arrives before $p_1$
    • $nic_{y+1}$ drops the out of order copy
    • duplicate will arrive

• The algorithm eventually updates the route when router gets two copies of a packet

• $[nic_i, nic_{i+1}]$ can be replaced by the multi-cast scheme iff $i_c < (i + 1)_c$

⇒ Implying no cycles are introduced
Control Clearance Packet Technique
\[(\text{nic}_x, \text{nic}_{x+1})_N\]

\[+ (\text{nic}_x, \text{nic}_{x+1}) - (s, d)\]

\[\text{nic}_{x+1}(s, d)\]

\[(\text{ctrl}, \text{ctrl})\]
Simulation Setup and Results
Simulation Setup

• Mininet – network emulator
• POX – OpenFlow controller
• Linux based OS as a VM guest
  • Core-i7 CPU, 4GB RAM
• Host machine - Core-i7 CPU, 16GB RAM
• Each simulation run was a clear execution
Simulation Results

• Simulation of
  • Sequential Replacement Algorithm
  • Control Clearance Replacement Technique

• Results Compared to the trivial algorithm:
  • Stops sending packets
  • Waits for all packets to arrive to d
  • Remove all relevant entries from the old route and establish a new route
  • Resume communication
Throughput during route update

The graph shows the throughput rate of packets per time unit over time units. The x-axis represents time units, and the y-axis represents the rate of packets per time unit.

Three algorithms are compared:

- Trivial Algorithm (red line)
- Sequential Replacement Algorithm (blue line)
- Control Clearance Replacement Technique (green line)

The graph indicates how each algorithm performs during route updates, with the Trivial Algorithm showing the highest initial throughput but also the largest drop during the update.
Conclusions

• This work presents a solution for routing consistency during updates

• Even without requiring the routers to identify the duplicated packets.
Thank You!