Consensus Routing:
The Internet as a Distributed System

Presented By
Naveen Cherukuri
Overview

- Introduction
- Consistency Issues
- Consensus Routing Overview
- Stable Mode
- Transient Mode
- Evaluation
Introduction

- Traditional Routing Protocols (BGP, OSPF, RIP)
  - Responsiveness (liveliness) over consistency (safety)
- Lack of consistency is at the root of bigger problems in Internet.
- BGP updates cause 30% packet loss and 90% of this loss is due to transient loops.
- Goal is to design a simple, practical routing protocol that allows general routing policies and achieves high availability.
Consistency Issues

BGP link failures

iBGP link recovery

BGP policy change

BGP policy cycles
Consensus Routing

- Safety: A router forwards a packet strictly along the path adopted by the upstream routers unless the adopted route encounters a failed link or router.
- Liveliness: The network reacts quickly to failures or policy changes and ensures high end-to-end availability.
- Key idea is to cleanly separate safety and liveliness concerns.
- Consistency is achieved using a distributed coordination algorithm among the routers.
- Packets are forwarded using two modes 1) Stable mode that only uses consistent routes and 2) Transient mode that heuristically forwards packets when a stable route is not available.
Stable Mode

- Update Log (Step-1)

Router State

**Routing Information Base (RIB)**

**History**

**Stable Forwarding Table (SFT)**

**Triggers:** (AS number, trigger number), when to generate a new trigger

- Link or next-hop router failed
- Local policy change
- A route from neighbor replaced current route.

**PROCESS_UPDATE**(\(B, r, t\)):
1. Add the update’s trigger \(t\) to the local set of incomplete triggers \(I_A\).
2. Process the update as in BGP. Let \(old\) and \(new\) be the best route to the prefix before and after the update. We define \(next\_hop\) for a route to be the first AS in the route.
3. Add the received update \((t, r)\) to the head of the History list. Consider the following cases:
   (a) \(old\_next\_hop\) is not \(B\), and \(new\_next\_hop\) is not \(B\): do nothing since the best route has not changed.
   (b) \(old\_next\_hop\) is not \(B\), and \(new\_next\_hop\) is \(B\): propagate \(new\) to neighbors with trigger \(t'\), where \(t'\) is a newly generated trigger. Add the selected update \((t', new)\) to the start of History.
   (c) \(old\_next\_hop\) is \(B\): propagate \(new\) to neighbors with unchanged trigger \(t\). Add the selected update \((t, new)\) to the start of History.
4. Remove \(t\) from \(I_A\).
• Distributed Snapshot (step-2)

An update can be incomplete at an AS when the snapshot is taken because 1) the update is in \( I_A \) 2) AS is waiting for MRAI timer to propagate the update 3) the update is in transit from neighboring AS.

**SNAPSHOT:**

1. Save the sequence of triggers in *History* as \( \overline{H}_A \).
2. Start logging any triggers received on channels other than the one on which the marker was received.
3. Initialize the set of incomplete triggers \( \overline{I}_A \) to \( \epsilon \). Add the set of triggers in \( I_A \) to \( \overline{I}_A \); these triggers correspond to the updates currently being processed.
4. Scan the outgoing queues for updates waiting on MRAI timers to expire, and add their triggers to \( \overline{I}_A \).
5. Send a marker to all neighbors.
6. Stop logging triggers on a channel upon receiving a marker on that channel.
7. Once the marker has been received on all channels, add logged triggers to \( \overline{I}_A \). These correspond to updates in transit during the snapshot.
• Frontier Computation (Step-3)
  
  ➢ Aggregation: Sending $\overline{H}_A$ and $\overline{I}_A$ to all consolidators.
  
  ➢ Consolidators – Tier-1 ASes
  
  ➢ Consensus: All consolidators reach consensus on Set of ASes $S$ and set of incomplete updates $I$.

  \[
  \text{COMPUTE.INCOMPLETE}(S, \overline{I}_A[], \overline{H}_A[]):
  \]
  1. Initialize $I = \bigcup_{A \in S} \overline{I}_A$.
  2. Do until $I$ reaches a fixed point:
     (a) For each $t \in I$, for each $A$ do:
        i. if $t$ occurs in $\overline{H}_A$, add the first occurrence of $t$ and all subsequent triggers in $\overline{H}_A$ to $I$.

  ➢ Flood: The consolidators flood the set of incomplete triggers $I$ and membership set $S$ to all ASes.
• Building Forward Table (Step-4)

\[
\text{BUILD\_SFT}(I, S):
\begin{enumerate}
\item Copy the current SFT to be its previous SFT.
\item For each destination prefix \( p \):
\begin{enumerate}
\item Find the latest \textit{selected update} \( u = (t, r) \) in \( p \)'s \textit{History} such that \( t \) is complete, i.e., neither \( t \) nor any preceding trigger is in \( I \).
\item Adopt \( r \) as the route to \( p \) in the new SFT.
\item Drop all records before \( u \) from \( p \)'s \textit{History}.
\end{enumerate}
\end{enumerate}
\]

• View Change

Maintains \( K^{th} \) and \( (K+1)^{th} \) SFT in epoch \( K+1 \).
Multiple Routers in an AS
- One or more routers act as local consolidators.

Protocol Robustness
- AS fails to send its snapshot in time: It will not be used for forwarding traffic in the next epoch.
- Consolidator fails: Distributed Consensus algorithm ensures consistent view.
- Recovery: Recovered AS will exchange paths with neighbors, compute SFT and participate at the end of the epoch.
Transient Mode

- Packet forwarding mode when stable route is not available at a router.
  - Routing Deflections
  - Detour Routing
  - Backup routes

- Implementation issues?
Evaluation

- Routing protocols effectiveness is studied in three cases:
  1) Link failures
  2) Traffic engineering accomplished by announcing and withdrawing sub-prefixes.
  3) Traffic engineering accomplished by AS path prepending.

- Consensus routing added about 8% in update processing overhead and about 11% additional lines of code to the BGP implementation.
Figure 6: Loops and disconnectivity in BGP following a failure.

Figure 7: Disconnectivity in consensus routing following a failure.

Figure 8: Traffic engineered subprefixes causes loops in BGP.

Figure 9: Path prepending causes intra-domain loops in BGP, leading to disconnectivity.
Overhead of Consensus routing

- Volume of Control Traffic

![Graph showing cumulative fraction of failure cases versus average number of bytes received by each AS for different epochs.]

Figure 10: Control traffic required by consensus routing.

- Cost of Consensus

<table>
<thead>
<tr>
<th>Number of nodes</th>
<th>Time when first node learns value</th>
<th>Time when last node learns value</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>434 ms</td>
<td>490 ms</td>
</tr>
<tr>
<td>18</td>
<td>485 ms</td>
<td>1355 ms</td>
</tr>
<tr>
<td>27</td>
<td>590 ms</td>
<td>1723 ms</td>
</tr>
</tbody>
</table>
• Path Dilation

![Graph of Path Dilation](image)

Figure 11: Path dilation incurred by interdomain transient mode options.

• Response Time

![Graph of Response Time](image)

Figure 12: Delay incurred by consensus routing between receiving and using a path.
Discussion

- Scalability of this approach?

- Handling Malicious AS to achieve consensus?

- Highly dynamic network topology – where links change rapidly between time of SFT and time of snapshot?
Thank you

Questions and Suggestions