Scalable Event Notification for MENTER

Kuo-Tung Kuo
Kevin J. Gallicchio
Bobby Bhattacharjee

University of Maryland
College Park, Maryland
Motivation

- MENTER goals:
  - Integrate traffic engineering and network management
  - Monitor and modify network-level properties at fine timescales
- MENTER deviates from existing approaches by reducing the feedback loop between monitoring and control
- However, need new feedback and control mechanisms
- **Active events** — Scalable, flexible messaging layer for MENTER
When is a scalable event notification system useful?
Why can’t we do this using existing protocols?

- Consider existing event notification protocols, e.g. SNMP
- **Traps** do exist for catastrophic events
- However, the polling architecture does not scale for fine grained events
- Difficult (impossible?) to properly configure infrastructure if specific event is not in a MIB
- Pins solutions to the centralized part of the soln. space
What can we do better?

- Define arbitrary, dynamically extensible atomic events, potentially on a per-flow/per-application basis
  - I-frame dropped event
- Combine events using customizable aggregation functions, and across nodes
  - \( x \) GOPs in a row not decodeable
- Custom event triggers based upon the evolution of variables
  - Latency for a premium flow monotonically increasing
- Subscription-based synthesis and forwarding of events to network-edge and centralized controller
  - Fetch the highest jitter on a large multicast group
Event Specification

- Need a simple, extensible format to specify and execute queries
- Use tuples of the form:

  \( \langle \text{when} : \text{what} : \text{where} \rangle \)

- \( \text{when} \in \{ \text{periodic, based upon node state} \} \)
- \( \text{where} \in \{ \text{to named node, to neighbors, up – down LSP} \} \)
- How to specify and query node state?
A little-language for node-state specification

- Need a system that is light-weight and relatively expressive
  - Do not need full Turing-complete language
  - The data-path must be preserved
- Current effort: extend work done at BBN in the **SmartPackets** project
A quick overview of SmartPackets

... and how it fits into MENTER

- SmartPackets consist of two languages: Sprocket and Spanner.
- Sprocket: C-like high level language (safe)
- Spanner: CISC-assembly for Sprocket
- Spanner is translated into a portable binary format and executed by a safe VM
- Sprocket and Spanner original design goals well aligned with our goals for MENTER
Example Operation
Node-level detail

- **User**
  - LSP setup
  - resource management
  - queries/replies

- **Kernel**
  - Label Switching Module
  - ALTQ Module

- **Network**
  - snd/rcv LSP queries

- **Network**
  - SmartPackets VM
  - demux

- **Module**
  - Label Switching Module
  - ALTQ Module

- **Resource Management**
  - LSP setup
  - resource management
  - queries/replies
Current Work

- Integrate and test NIST/SmartPacket and our own software
- Testbed ready for defining customizable events
  Several LSP-specific events have been defined and tested
- Leverage prior work for event transport
Research Challenges

- Define useful atomic and aggregate events in conjunction with simulation work
- Analyze safety and stability properties for in-network distributed control