A Two-Level Architecture for Internet Signaling

Bob Braden
USC Information Sciences Institute

IETF 54
July 15, 2002
A Two-Level Architecture for Signaling

• draft-braden-2level-signaling-00.txt, Nov 2001
  ~http://www.isi.edu/rsvp/ site or
  ~http://www.isi.edu/~braden/draft-braden-2level-signaling-00.txt

• Intended as a partial design for “RSVP v2”
  – i.e., an RSVP-like signaling protocol for a much broader range of
    signaling applications than the original RSVP design goal.
What IS “Signaling”?

• Broadest definition: *any flow of control info*; e.g.:
  – QoS set up (RFC 2205: RSVP v1)
  – Network provisioning
  – Middle box control (e.g., firewalls)
  – IGMP (ES -> 1st hop router state)
  – DHCP (Server -> ES state)
  – TCP ACKs and Window updates

• More useful definition: setting up per-flow state in routers (and middle boxes?) along the path of a data flow.
  – “Path-directed”
  – Not necessarily micro-flows, could be aggregated flows.
  – May be initiated by an ES or by a network control entity
What is the Signaling Problem?

- There are many different signaling problems.
- This is a complex problem space; hard to figure out how to partition it effectively and to use common mechanisms. Mapping and partitioning this space is the task of this WG.
- Recommend: follow the old-time Internet religion...
  - Cope with heterogeneity
  - Cope with failures
  - General solutions and common mechanisms, to the extent possible.
  - Pay careful attention to modularity.
  - Provide for application-specific engineering within the broader design.
What can we learn from RSVP V1?

- RSVP maintains the Internet architectural distinction between IP layer (E2E) and the link layer. Much complexity can be “hidden” in the LL; IP provides the E2E glue.
  - Each subnetwork may have its own signaling -- e.g., ATM, or the Subnet Bandwidth Manager (SBM) (adapted RSVP for 802 QoS signaling)

A crazy idea: maybe the tension between mobility signaling and E2E signaling in NSIS could be clarified if the mobility mechanism could be treated as if it were a link layer for the purposes of signaling. Maybe the strict IP/LL protocol boundary is not the right signaling abstraction boundary in this case.
What can we learn from RSVP V1? (2)

• RSVP V1 came to the IETF with the basic requirements and protocol design already published, simulated, and prototyped. And then it took 3 years...

• RSVP is adaptable to a wide variety of signaling problems... “RSVP-like”:
  • MPLS Path setup
  • VPN provisioning
  • Optical network configuration
  • Link-layer QOS setup
  • TIST -- Topology-Insensitive Service Protocol (Middle boxes)
  • AFSP -- Active Filter Signaling Protocol -- setting filters for efficient multicast publish/subscribe system (US DoD distributed simulations)
What Can We Learn from RSVP V1?

• RSVP V1 is adaptable both:
  – syntactically \( (TLV\ text{\ encoding}) \) and
  – semantically \( (path-directed\ signaling\ mechanism) \)

• But it does not fit together well ... Confusion, overlap, conflicts among features and signaling models. What IS a conforming RSVP v1 implementation today??

• Can we do better?
Next Step in RSVP-like Signaling

• Ideal: modular building blocks that can be assembled for each particular signaling application.
  • Note strong analogy to reliable multicast problem!
    – But that’s hard, harder than I want to think about now.

• Simpler: separate the common stuff from the (signaling) application-specific stuff.

=> Two-level signaling architecture
  □ ALSP -- Application-Level Signaling Protocol (many of these)
  □ CSTP -- Common Signaling Transport Protocol
Two-Level Model

• Functional Partition:
  
  – ALSP has all the E-to-E semantics

  \[ \text{P-src} \xrightarrow{\text{Data/signaling path}} \text{P-sink} \]

  – CSTP handles only hop-by-hop semantics

  \[ \text{H-src} \xrightarrow{xSIG(SAPU,...)} \text{H-sink} \]

  SAPU = Signaling App Protocol Unit

  Neighbor nodes
## Functional Partition

<table>
<thead>
<tr>
<th>CSTP</th>
<th>ALSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Reliable ordered delivery* (trigger messages)</td>
<td>(To emulate RSVP V1:)</td>
</tr>
<tr>
<td>• Soft-state refresh*</td>
<td>• QoS (i.e., flowspecs, etc)</td>
</tr>
<tr>
<td>• Fragment/reasm SADUs*</td>
<td>• Merging</td>
</tr>
<tr>
<td>• Unicast &amp; multicast</td>
<td>• Styles</td>
</tr>
<tr>
<td>• H/H security &amp; IPsec</td>
<td>• Receiver-orientation</td>
</tr>
<tr>
<td>• Congestion control</td>
<td><strong>Much of the detail, and most of the complexity, of RFC 2205.</strong></td>
</tr>
</tbody>
</table>

*Optional (avoid OSI disease because ALSP => profile.)

*Much adapted from RFC 2961.*
Read the I-D for Details...

Defines:

• **ALSP/CSTP API**
  – E.g., ALSP can call:
    
    ```
    sendNewSAPU( SAPU, IP-targ, [OIF]) -> SAPUid
    
    IP-targ = P-sink or H-sink
    OIF = outgoing interface (for multicast)
    ```

• **Contents of CSTP messages**
  – CSTP sends to IP-targ address:
    
    ```
    xSIG( NEW, h-src, SAPUid, SAPU)
    ```
  
  – CSTP retransmits until it receives:
    
    ```
    xSIG( ACK, SAPUid)
    ```
  
  – CSTP may then send periodic refreshes:
    
    ```
    xSIG( REFRESH, h-src, SAPUid)
    ```
Conclusion

• The two-level model provides useful functional modularity.

• It could also provide a blueprint for organizing IETF working groups to engineer a variety of signaling applications.

• Writing a Requirements document (IMHO) is simply an exercise in frustration without some initial architectural framework. I suggest using something like my two-level architecture as this framework, with also some thought about the link layer/IP layer abstraction boundary.

• In any case, you need SOME architectural framework.
A Modest Proposal for NSIS

1. Define the architectural framework

2. Map the signaling problem space into a rational set of ALSPs.

3. Charter a Signaling WG to define CSTP and one ALSP.
   
   Successive refinement on CSTP/ALSP interface and CSTP functionality needs close coupling, should be in same WG.

4. Later, other WGs can define additional ALSPs.