Efficient Cross-Layer Negotiation

Bryan Ford
MPI-SWS and
Yale University

Janardhan Iyengar
Franklin & Marshall College

Presented at HotNets-VIII, October 22, 2009

“Tng: Transport Next Generation” Project
Support: NSF FIND grants CNS-0916413 and CNS-0916678
A Proliferation of Layers and Layer Combinations

Application
- HTTP
- FTP
- DNS
- RTP

Transport Security
- SSL
- DTLS

Transport
- SCTP
- TCP
- UDP
- DCCP

Network Security
- IPsec
- IPv6
- UDP
- Teredo
- IPsec
- IPv6
- UDP

Network
- IP
- IPv6

Data Link
- Ethernet
- Token-Ring
- PPP

(DirectAccess)
- IP
- Ethernet
Future: Ever More Layers/Combinations?

**Multi-Streaming Transports**
- SCTP [rfc4960], SST [SIGCOMM '07]

**Multipath Transports**
- SCTP [rfc4960], MPTCP [WIP]

**Further Decomposition**
- “Breaking Up the Transport Logjam”, HotNets '08

---

**Application**

<table>
<thead>
<tr>
<th>Stream</th>
<th>Stream</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel</td>
<td></td>
</tr>
<tr>
<td>Network</td>
<td></td>
</tr>
<tr>
<td>Link</td>
<td></td>
</tr>
</tbody>
</table>

**Application**

<table>
<thead>
<tr>
<th>Multipath Transport</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subflow</td>
</tr>
<tr>
<td>Network</td>
</tr>
<tr>
<td>Link</td>
</tr>
</tbody>
</table>

**Application**

<table>
<thead>
<tr>
<th>Semantic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Isolation</td>
</tr>
<tr>
<td>Flow</td>
</tr>
<tr>
<td>Endpoint</td>
</tr>
<tr>
<td>Network</td>
</tr>
<tr>
<td>Link</td>
</tr>
</tbody>
</table>
The Negotiation Problem

Decisions, decisions!

Application

Transport

Security

Transport

Network

IPv4
IPv6

TCP
SCTP

SSL

HTTP

SIP
IAX

UDP
DCCP

DTLS

IPv4
IPv6
Compatibility and Preference

Which combinations do *both* endpoints support?
Which combinations do they *prefer*?
Talk Outline

• Background and Alternatives
• A Model for Negotiation
• Negotiation Transport Protocol
• Discussion, Conclusion
Background and Alternatives
Approach 0: Name Encoding

http:// means:
  HTTP
  TCP
  IP
  [rfc2616]

https:// means:
  HTTP
  SSL
  TCP
  IP
  [rfc2818]

http++sctp:// means:
  HTTP
  SCTP
  IP
  [draft-wood-tae-specifying-uri-transports]

http++ssl++sctp:// means:
  HTTP
  SSL
  SCTP
  IP
  ?
Disadvantages of Name Encoding

**Loss of Transparency**
- User cares about *application*, not underlying stack...
  but is *forced* to see and care about underlying stack
- When underlying stack changes, URLs change/break
  - redirectors proliferate between `http://` and `https://` spaces

**Loss of Compatibility**
- If user puts “`http++sctp://...`” link on a web page,
  legacy browsers break; *cannot* fall back to TCP

**Where Do You Stop?**
- “`http++tls++tcp++ipv6++ethernet`” ???
Approach 1: Try and Fall Back
Challenge 1: Controlling Delay

- Failures can incur *timeouts* (e.g., due to NATs)
- ...potentially *compounded* by layering
Approach 2: Try in Parallel

Host A

SCTP INIT
TCP INIT

Host B

SCTP RST
TCP ACK
Challenge 2a: Redundant State

Host A

SCTP INIT

TCP INIT

Host B

SCTP ACK

TCP ACK
Challenge 2b: Combinations

Layering can lead to explosion of choices
Approach 3: Out-of-Band Information
Challenge 3a: Administration

DNS server must know:

- Name→IP mapping (as before)
- Entire protocol stack supported by Host B
- Protocol options...?

⇒ *Synchronization Nightmare?*
Challenge 3b: E2E Robustness

If endpoints agree on configuration X, will it work?
Our Solution: **Negotiation**

- Hosts explicitly describe possible configurations during initial “meta-communication” exchange, **before** actual communication commences.

---

**Host A**

```
“Hi, I speak:
- IPv4
- IPv6
- UDP
- DCCP
- DTLS
- SIP
- IAX
```

**Host B**

```
“Hi, I speak:
- IPv4
- IPv6
- UDP
- DCCP
- DTLS
- SIP
- IAX
```
A Model for Negotiation
Negotiation Model Overview

1. Initiator sends a **Protocol Graph Proposal**
2. Responder returns **Revised Protocol Graph**
3. (Optional) further protocol graph revision steps
4. Peers commit, **Acknowledge Protocol Graph**
5. Peers communicate via negotiated protocols
Message 1: Initiator → Responder: Propose Protocol Graph

Negotiation Message 1

Host A

TCP

TLS

DTLS

DCCP

base (IP)

(opt1 | opt2)

(opt1 | opt2)

(opt1 | opt2)

(opt1 | opt2)

(goal (SIP)

(alternatives)

Host B
Message 2: Responder → Initiator: Revise Protocol Graph

Negotiation Message 2

Host A

TCP

TLS

DCCP

base (IP)

goal (SIP)

opt1

opt2

Host B

opt1

opt2

opt1

opt2

opt1

opt2

opt1

opt2

opt1

opt2
Message 3: Initiator → Responder:

Acknowledge Protocol Graph

Host A

goal (SIP)

opt2

TLS

opt1

TCP

base (IP)

Host B
Message 4+:
According to Negotiated Stack

Host A

Normal Packets

SIP
TLS
TCP

Host B
Concurrent Protocol Initialization

Whenever feasible:

- **embed** protocol-specific handshake info into graph
- **run handshakes concurrently** while negotiating
- **commit** only negotiated configuration atomically

```
HOST A

1. SIP REGISTER
   - TLS ClientHello
   - DTLS ClientHello
   - TCP INIT
   - DCCP Request

2. SIP 200 OK
   - TLS ServerHello
   - DTLS ServerHello
   - TCP INIT-ACK
   - DCCP Reply
```

Host A
Key Benefits of Negotiation Model

- Supports backward-compatible evolution
  - New smart nodes can fall back on old dumb protocols
- Happens strictly between nodes concerned
  - Users don't have to care (e.g., between http & https)
  - Name server administrators don't have to care
- Protocol graph representation scales to handle:
  - Arbitrarily deep protocol stacks
  - Many alternatives per layer
- Setup whole “layer cakes” in minimal # of RTTs
  - Regardless of protocol stack depth
Further Challenges & Extensions

(see paper)

● Multi-Round Negotiation
  – due to dependencies, hiding of alternatives, graph size

● Negotiation Across Multiple Contexts
  – IPv4 vs IPv6, new protocol vs legacy, UDP encapsulation

● Recursive Negotiation
  – negotiate “crypto wrapper” and “contents” concurrently

● Peer-to-Peer Negotiation
  – symmetric peers must converge on a configuration
Negotiation
Transport
Protocol
How to Express Protocol Graphs?

Negotiation Message Structure:

<table>
<thead>
<tr>
<th>Negotiation Message</th>
<th>Node Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node #1</td>
<td>Num Children</td>
</tr>
<tr>
<td></td>
<td>Options Length</td>
</tr>
<tr>
<td>Node #2</td>
<td>Child 1 Node ID</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Node #n</td>
<td>Child m Node ID</td>
</tr>
</tbody>
</table>

Negotiation Options (variable)

Protocol-Specific Data (variable)
How to Convey Protocol Graphs?

Negotiation messages might be big:
- Many layers × many alternatives for each to describe
- Embedded protocol-specific data: crypto keys, etc.

Individual graph nodes may be large or small
- Segment large nodes, aggregate small ones into packets

Receiver probably wants only specific nodes
- Efficiently ignore/drop anything it doesn't understand

⇒ Specialized Negotiation Transport Protocol
Negotiation Transport: Packet Structure

Fixed header + multiple chunks [SCTP] each describing different graph node
Negotiation Transport

Negotiation packet sequencing permits individual packet ack/retransmit [SST]

<table>
<thead>
<tr>
<th>Msg Type</th>
<th>Step Number</th>
<th>Transmit Seq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Negotiation Protocol Magic Cookie</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Negotiation Transaction ID</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>AckCt</td>
<td>Ack Seq</td>
</tr>
</tbody>
</table>
Negotiation Transport

Each chunk describes [part of] a graph node

- Receiver can ack & discard *all* chunks for unknown protocols without storing *any*
Not needed

Let negotiated protocol worry about:

- Connection state machines
- Application-friendly semantics (e.g., streams)
- Flow control
- Congestion control (beyond slow-start)
- ...

...
Discussion, Conclusion
What Doesn't (Really) Work

- Encoding protocol stacks in names
  - Non-transparent to user; compatibility hell
- Try alternatives serially & fall back
  - Delay/timeout hell
- Probe alternatives in parallel
  - Redundant protocol instances; combinatorial hell
- Encode alternatives in DNS responses
  - Not end-to-end robust; administrative hell
What *Might* Work

Explicit In-Band Negotiation:

- Get user & third parties out of the loop
- Describe alternatives in compact protocol graphs
- Handshake deep layer cakes concurrently
- Receiver stores only what he understands & wants

“*Tng*: Transport Next Generation” project

[http://bford.info/tng/](http://bford.info/tng/)

Support: NSF FIND grants CNS-0916678 and CNS-0916413