Application Driven Networking with XSP

Ezra Kissel, Martin Swany
Indiana University, InCNTRE
Introduction

• Software Defined Networking and OpenFlow
  – Fine-grained control of forwarding in the data plane
  – Tremendous interest in R&E as well as industry

• Network as a service, dynamic links/circuits
  – Network applications can explicitly allocate for efficiency or determinism
  – Cloud-based infrastructure: networks complete the picture
  – Good for the network to herd the “elephant” flows

• “Northbound” interface for SDN
  – How do applications interact with services in the network
  – Support traffic engineered flows (e.g. bulk data movement), etc.

• XSP: session-based protocol framework for interaction between applications and network services
eXtensible Session Protocol (XSP)

• In the spirit of the ITU-T Recommendation X.225 connection-oriented session protocol specification:
  – “… a single protocol for the transfer of data and control information from one session entity to a peer session entity... making use of the services available from the Transport Layer”

• XSP resides above the transport layer
  – Can encapsulate control and data PDUs into session layer PDUs
  – Inline and ongoing unlike SIP

• Provides a common set of features useful to applications, rather than being part of each app
  – Bidirectional communication between apps and network svcs

• Service interaction is session in the most literal sense:
  – “a period of time devoted to a particular activity”
What’s in a session layer?

• End to end state
  – Session state for related parallel or serial transport connections
  – Context for mobility, multipath

• Authorization and authentication
  – X.509 or SSH credentials

• Naming and addressing
  – Session `connect()` not tied to network or transport addresses

• Explicit signaling and data channel optimizations
  – Phoebus WAN optimization

• Flexibility for new approaches and technologies
  – SDN
  – Delay tolerant flows for performance (buffer and burst)
XSP architecture and library support

- Modular framework
  - Distinct “service handlers” (SH)
  - Security, Path, Protocol Channel, etc.

- XSP library: \textit{libXSP}

- XSP-enabled services link with \textit{libXSP}
  - Register new functionality via library hooks
  - Provide serialization / deserialization methods

- Client library for applications
  - Sockets-like interface
  - Transparent wrapper (shim library)

- XSPd (daemon)
  - Implements protocol frontend
  - Platform for development/prototyping
XSP as a network configuration interface

- **Path** framework
  - Modular set of service handlers for different technologies
  - Esnet OSCARS, OpenFlow, …

- A path is a set of rules applied in the network

- API provides a consistent abstraction
  - Access to technology-specific fields if needed

- Service called XSPd-SDN for SDN deployments
XSP and SDNs via OpenFlow

• Avoid “reactive” installation of rules
  – Potential issues with scalability and policy across domains

• XSP enables a proactive model
  – Application-driven

• Build a path across domains with user-provided credentials

• XSPd-SDN can support a hybrid approach
  – Reactive in data centers
  – Proactive for traffic engineered paths
  – Can talk to other provisioning services
Use Case: Building end-to-end virtual paths

- Globus GridFTP transfer application
  - Considered the “gold standard” in Grid community for bulk data movement
- XSP driver developed within Globus XIO framework allows clients to signal the network and explicitly use network-based services
- SC11 demo with ESnet using ECSEL and OSCARS

```
globus-url-copy -vb -p 4 -dcstack
xsp:"xsp_hop=<XSPd-SDN>/5006;xsp_net_path=<TYPE>" , phoebus:"phoebus_path=<GW1>/5006#<GW2>/5006 "
```
Use Case: Application-driven path selection

- Dynamically signal path selection based on:
  - Explicit threshold
  - In progress: direct network feedback

- SC11 demonstration of XSPd-SDN
  - GridFTP XSP driver with active performance threshold
  - Two domains (IU and SC11-Seattle)
  - Shared 100G and dedicated 10G path
  - 10G connected end-hosts
  - Automatic performance improvements
Use Case: Dynamic firewall

• “Default drop” router with explicit authentication
  – Prototype using JunOS SDK on MX
  – XSPd-SDN running on routing engine
  – On-demand installation of forwarding rules

• Bypass conventional firewall
  – Support high-performance flows
  – Explicit authentication installs rules for bypass
Summary

• XSP is a session-layer protocol specification and framework for the exchange of control and data between applications and in-the-network services

• Provides an interface for configuring advanced networks
  – SDNs including (but not limited to) those implemented with OpenFlow
  – “Network as a service”

• Includes extensible AuthN/AuthZ
  – SSL/X.509, SSH, password-based, etc.

• Remains flexible for future improvements
  – Easy to implement and integrate new handlers

• Demonstrated to work in a number of application-driven scenarios
Thank you for your time

Thanks to our colleagues at IU, ESnet and Internet2
Support: NSF OCI-0910812, OCI-1127349, and CNS-105011, DOE DE-SC0001421

Stay tuned for the demo!

http://damsi.cs.indiana.edu/projects/phoebus
Session Layer Related Work


