An Architectural Perspective on Multipath Transport

draft-ford-mptcp-architecture-00

The Higher-Order Bit

- Many high-level decisions are, or can be, bigger than MPTCP and apply to any multipath transport
  - Capture these where appropriate
- Lay out the design space for multipath transport
  - Goals and considerations
- Finally, show how the MPTCP proposals fit into this multipath transport architecture
  - Split high-level MPTCP design from details
  - Map MPTCP drafts to architecture

Goals of a Multipath Transport Architecture Document

(1) To identify functional and performance goals for a multipath transport;

(2) To describe necessary functional decomposition of transport layer to meet the above goals;

(3) To discuss protocol design considerations for the different components;

(4) To discuss interfacing among components and implementation suggestions;

(5) To discuss how the MPTCP drafts fit in this architectural framework

(1a) Identify Functional Goals For Multipath Transport

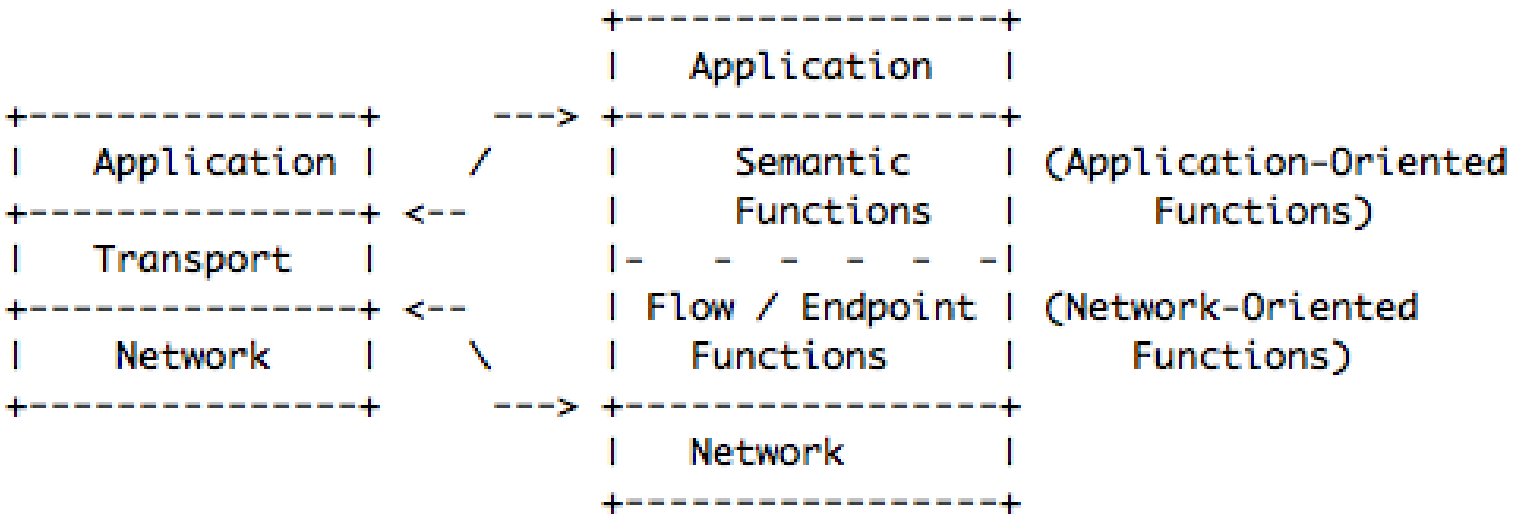
- Multihoming
  - Supporting hosts with multiple interfaces
- Application Compatibility
  - Multipath variants of existing transports should provide multipath capability for legacy apps without changing the service model
- Network Compatibility
  - With Internet as is, including middleboxes
- E2E Reliability and Security (across multiple paths)
- Automatic Negotiation (with fallback to legacy non-multipath variant)

(1b) Identify Performance/Efficiency Goals For Multipath Transport

- Resource Pooling
  - Optimizing network utility though shifting load away from congested bottlenecks to spare capacity
- TCP-Friendliness
  - Coexist gracefully with existing transport flows
- Congestion State Sharing
  - Across multiple flows within an app and/or across multiple apps
- Small Transaction support
  - Bulk transport is not the only use case; minimize multipath overhead

(2) Functional Decomposition Of Transport Layer To Achieve Goals

- Network-oriented **Flow/Endpoint** functions
  - of interest to middleboxes (endpoints (addresses, ports); congestion control)
- Application-oriented **Semantic** functions
  - of interest to applications (reliability, ordering, ...)



- A new location for security functions: between the two functional components

(3) Discuss Protocol Design Considerations For Different Components

- Semantic Component:
  - e2e reliability, security across multiple "flows"
  - transmission/retransmission policies (considerations for small files)
  - lightweight semantic "streams"
- Flow/Endpoint Component:
  - congestion control considerations (CC state sharing, resource pooling, PEP interactions, etc.)
  - "endpoint" identification considerations (multiple vs. single port number, NAT interactions)

(4) Discuss Interfaces Among Components and Implementation Suggestions

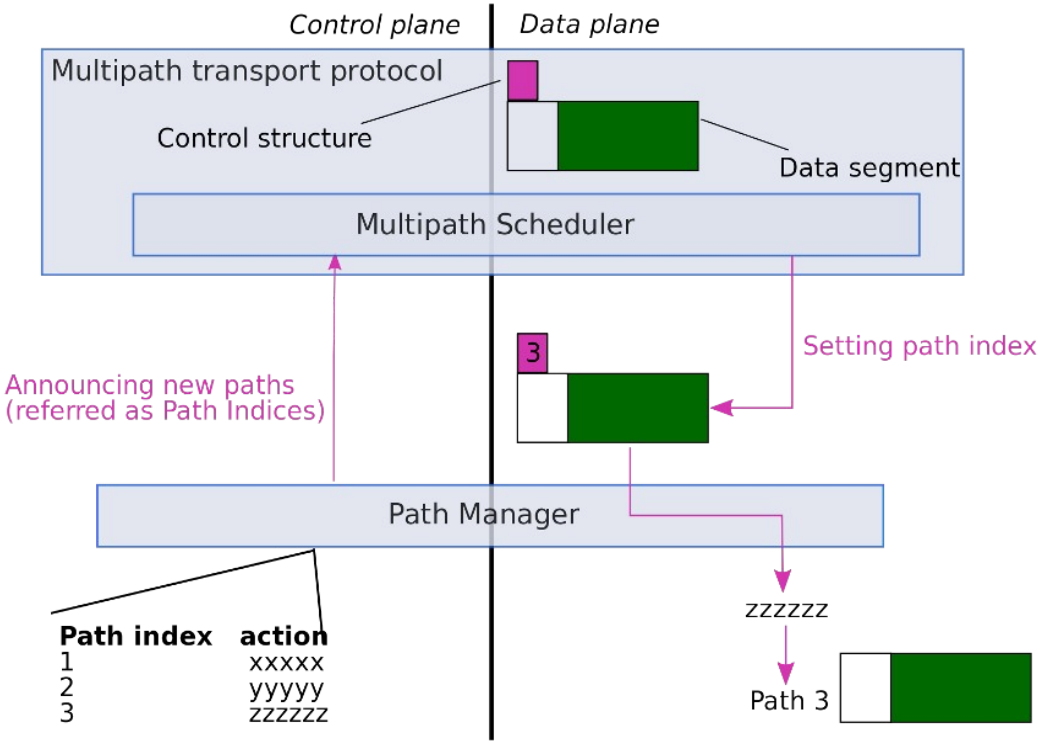
- Information flow between Semantic and Flow components for CC bundling / CC state sharing
  - Semantic layer needs to know about multiple flows, and pass data to appropriate flow
  - Path info (cwnd, RTT)
  - Others?
- Implementation suggestions
  - MPS / PM architecture and experience
  - Others?

MPS/PM Implementation Architecture

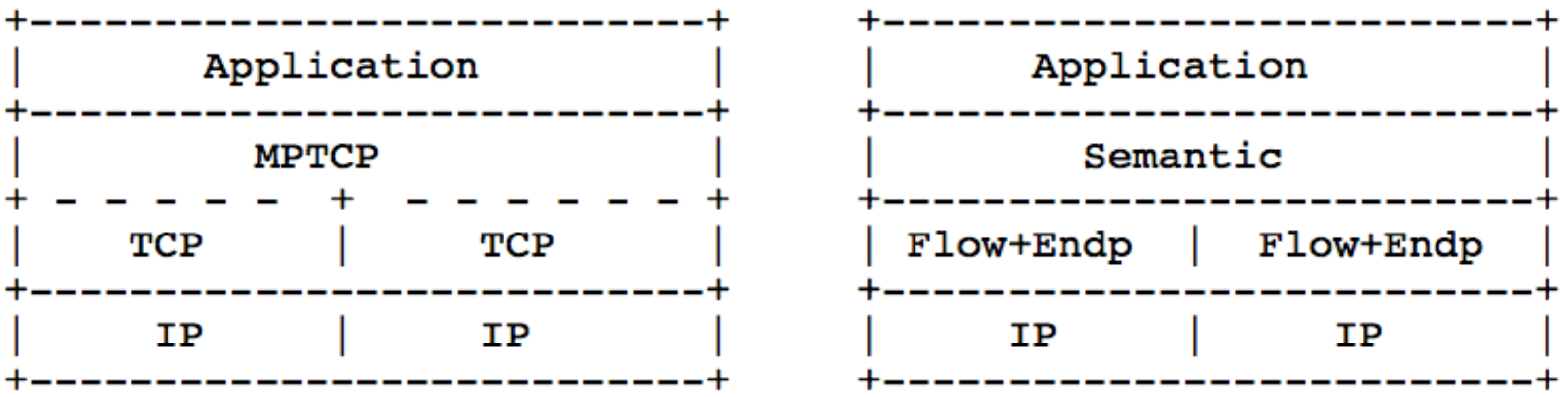
- Path Manager (PM)
  - Maps to Flow/Endpoint Layer
  - Discovers available paths and provide interface to them (via path indexes as an abstraction)
  - Handle necessary functions to use paths (e.g. using appropriate address for path)
- Multi-path Scheduler (MPS)
  - Maps to Semantic Layer
  - Receives data from application and encapsulates it appropriately for transmission
  - Decides which paths to use for each packet

Example MPS/PM Interface

- Internal architecture, with path announcements and using control structures to indicate between components what do do with data packets



(5) Discuss how MPTCP drafts fit in this multipath arch framework



- Maps MPTCP as Semantic, TCP as Flow/Endpoint
- Discuss architectural goals met and those not met
- How should an extended API influence the components?
- What does security protect and where should it fit?
- Others?

High-level MPTCP Design in the Architecture

- High-level design decisions take the architecture to the next step towards specification/implementation
- Identifies the bounds for a multipath-TCP design to work within
- High-level design decisions, once resolved:
  - Can be mapped to the architectural separation
  - Can be verified against the architectural goals

High-level MPTCP Design Decisions

From recent mailing list discussion (not exhaustive list)

- Protocol-related decisions
  - e.g. IP addresses used, initiators of subflows
- Congestion control algorithm
  - e.g. as good as TCP on best subflow
- API
  - e.g. no changes required, but extended API optional
- Security
  - e.g. mechanisms do not interfere with middleboxes

Where next?

- This work can be separated into:
  - Generic multipath transport architecture
  - High-level design decisions for a multipath TCP
  - Analysis of multipath TCP drafts' detailed design against architectural goals and high-level design

- Please provide feedback on:
  - Goals (and structure) for the document
  - Is the draft an appropriate start for this work?

Domo Arigato!