Jain *et al.*, "B4: Experience with a Globally-Deployed Software Defined WAN," *Proc. of ACM SIGCOMM '13*, 43(4):3-14, Aug. 2013.

Taeju Park
Ming zhi Yu

Overview

- Background introduction
- Integration of routing service
- Traffic engineering of B4
- Evaluation methods and results
- Novel points
- Improvements

WAN

- Computer networking technologies used to transmit data over long distances[1]
- Characteristics of traditional WAN architecture:
  - Links are expensive
  - Routers are expensive: place a premium on high availability
  - Overprovisioning: utilization is provisioned to 30% to 40% to protect against failures and packet loss
  - Typically, all bits are treated the same: if some links fail, it is unable to prioritize traffic that is sensitive to latency

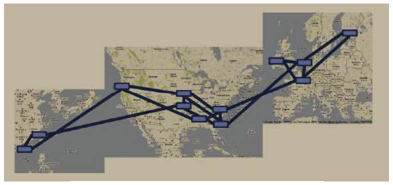


Figure 1: B4 worldwide deployment (2011)

Google's WANs

- Two distinct WANs
  - User-facing WAN: peers and exchange traffic with other Internet domains
  - B4: provide connectivity among data centers
- Usages of B4 (volume ↓, Latency sensitivity ↑)
  - User data copies (e.g. email, documents, audio/video files) to remote data centers
  - Remote storage access for computation over distributed data sources
  - Large-scale data push used to synchronize state across multiple data centers
- Characteristics of B4
  - Elastic bandwidth demands: majority of traffic can tolerate temporary bandwidth reductions
  - Moderate number of sites
  - Control over end applications: enforce application priorities
  - Cost sensitivity: overprovisioning is unsustainable due to capacity target and growth rate
- Traditional WAN architecture won't work for B4

Why SDN?

- Benefits of the separation of control plane and data plane
  - Vastly simplify coordination and orchestration for network changes
  - Can upgrade server independently from the switch hardware
  - Software and hardware can evolve independently
    - Control plane software becomes simpler
    - Data plane hardware becomes more programmable and has higher performance
- Testing environment is simplified
  - Emulated an entire software stack in a local cluster
- Enable rapid iteration on novel protocols
- A fabric-centric WAN view simplifies management

Overview of B4's Architecture

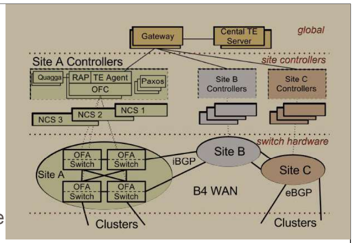


Figure 2: B4's Architecture

1. Global layer
   - Central TE Server: perform traffic engineering
   - Gateway: abstract details of OpenFlow and switch hardware
2. Site controller layer
   - NCS: network control servers which host OFCs and network control applications
   - Quagga: routing software suite which provides implementations of various routing algorithms
   - OFC (OpenFlow controller): an SDN controller that uses the OpenFlow protocol
   - Paxos: a family of protocols to handle leader election for fault tolerance
   - RAP (routing application proxy): provide connectivity between Quagga and switches
3. Switch hardware layer
   - Consists of switches. OFA running on Linux.
   - Primarily forwards traffic, does not run complex control software.

Integration of standard routing services

- Goal is to support hybrid network deployments
  - Standard routing services: BGP/ISIS
  - Traffic engineering
- Quagga was used to implement these standard routing services
  - Problem is that it has no data-plane connectivity because of OFC, which does have connectivity with switches
  - Developed a SDN application called Routing Application Proxy (RAP), to
    - provide connectivity between Quagga and switches
      - e.g. BGP/ISIS route updates, routing-protocol packets, switches' interface updates
    - translate each RIB entry into two OpenFlow tables

Integration of standard routing services

- ECMP Group table
  - ECMP: Equal-cost multi-path
  - Used to perform per-flow load balancing and enable the topology abstraction
- Flow table
  - Map prefixes to entry in a ECMP Group table

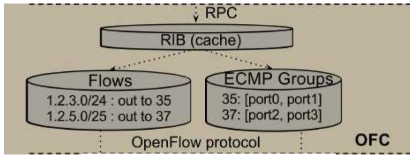
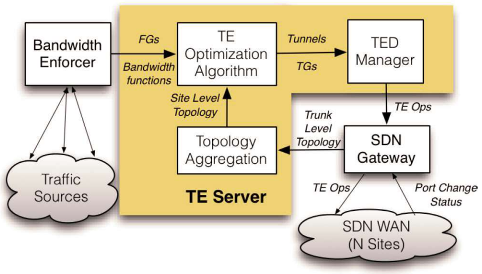


Figure 3: OFC

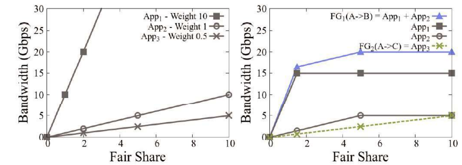
Centralized Traffic Engineering Architecture



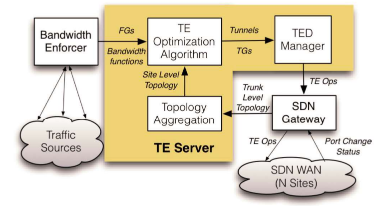
TE Server operates over network states

- Network Topology
  - Vertices: site
  - Edge: site-site connectivity
- Flow Group (FG)
  - Aggregation of applications
  - {source, destination, QoS}
- Tunnel(T)
  - Site-level path (A->B->C)
- Tunnel Group (TG)
  - Maps FGs to a set of tunnels and corresponding weights

Bandwidth Functions



(a) Per-application. (b) FG-level composition.



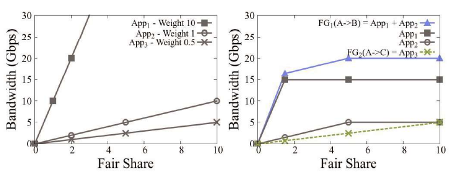
Bandwidth function

- Bandwidth allocation to an application
- Based on administrator-specified static weights (priority)

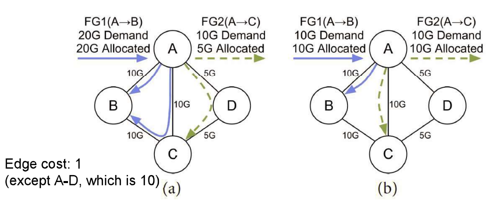
Bandwidth Enforcer

- Configure and measure bandwidth functions
- Provide bandwidth functions to TE server

TE Optimization Algorithm



(a) Per-application. (b) FG-level composition.



(a) (b)

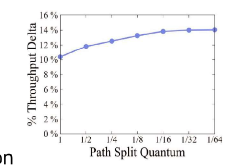
Tunnel Group Generation

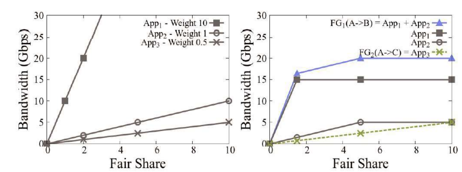
- Allocate bandwidth to FGs based on demand and priority.
- All competing FGs receive equal ***fair share***
- Preferred tunnel for a FG is the minimum cost path that does not include a bottleneck edge.

Example

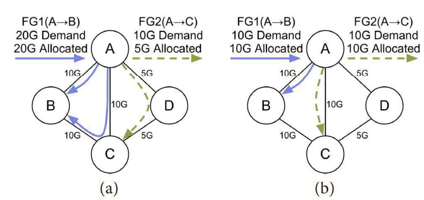
- FG1 (Split ratio; 0.5:0.4:0.1)
  - A->B (at 0.9): about 10Gbps
  - A->C->B (at 3.33): about 8.33 Gbps
  - A->D->C->B: 1.67 Gbps (fully Satisfied)
- FG2 (Split ratio; 0.3:0.7)
  - A->C (at 0.9): 0.45Gbps
  - A->C (at 3.33): 1.22Gbps
  - A->D->C: 3.33 Gbps (remaining)

TE Optimization Algorithm





(a) Per-application. (b) FG-level composition.



(a) (b)

Tunnel Group Quantization

- Adjust splits to the granularity supported by the underlying hardware
- Use greedy approach to find optimal split
  - Down quantize its split ratio (Rounding)
  - Add the remaining quantas to the available tunnels to make the solution max-min fair

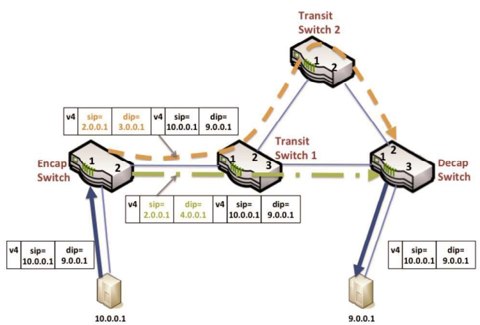
Example

Quanta : multiple of 0.5

- FG1 (0.5:0.4:0.1)
  - Down quantize -> (0.5:0.0:0.0)
  - Add remaining -> (0.5:0.5:0.0)
- FG2 (0.3:0.7)
  - Down quantize -> (0.0: 0.5)
  - Add remaining -> (0.0:1.0)

TE Protocol

- B4 switches operate in three roles



Multipath WAN Forwarding Example

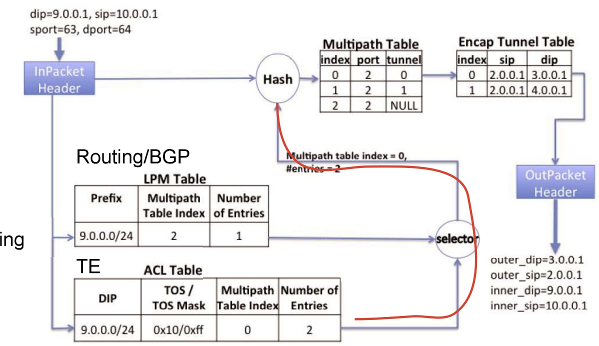
Encapsulating Switch
- Initiates tunnels
- Splits traffic between tunnels based on hash of the packet header

Transit Switch
- Forward packets based on destination IP (tunnel ID)

Decapsulating Switch
- Terminates tunnels
- Forward packets using regular routes

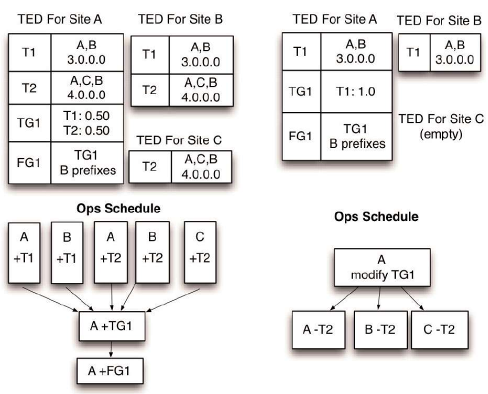
Composing routing and TE

Shortest-path routing
+
TE



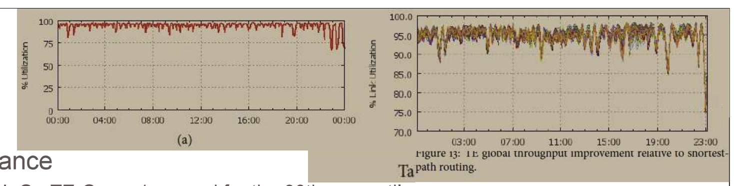
- Set its desired forwarding behavior
- Take precedence over LPM

Coordinating TE State Across Sites



- TE server coordinates T/TG/FG rule installation across multiple OFCs.
- TE Optimization output to Traffic Engineering Database (TED), needed to forward packets.
- Key-Value datastore for T, TGs, and FGs
- OFC converts TE op to flow-programming instructions.

Evaluation



- TE ops performance
  - Latency for a NoOp TE-Op → 1 second for the 99th percentile
  - Switch fraction time (STF = Switch time / Overall TE op time) → substantial, have potential for optimizations at lower layer
- Impact of failures. Measured the duration of any packet loss after six types of events:
  - a single link failure, an encap switch failure and separately the failure of its neighboring transit router, an OFC failover, a TE server failover, and disabling/enabling TE.
- TE algorithm. Measured how throughput varies with respect to number of path splits
- Link utilization and hashing
  - Edge level: utilization over a 24-hour period; ratio of high priority to low priority packets.
  - Link level: utilization over a 24-hour period; max:min ratio in link utilization

What we like about the paper and novel points

- The strategy they used to deploy B4
  - first deployed standard routing service then deployed TE
  - Gave time to develop and debug the SDN architecture before trying new features
- The way they prepare for potential failures
  - Support of hybrid network, both shortest path and TE
  - layered traffic engineering on top of baseline routing protocols provide a fail-safe mechanism
- Simplify complicated problem through abstraction
  - When designing traffic engineering, they abstract away the multiple links corresponding to one edge and use ECMP to enforce such abstraction
  - Significantly reduces the size of graph input to TE algorithm

What can be improved or extended?

- What is overlooked?
  - One underlying assumption in this paper is that a bandwidth function can be obtained for each applications. However, they didn't provide details about how to determine those bandwidth function (e.g. how to set the ratio between weights of a high priority application and a low priority application)
  - Admitted that human errors are responsible for most of the system failures, but didn't present a solution to automate system operations

References

[1] Wide area network [online]. Available:https://en.wikipedia.org/wiki/Wide_area_network