A Network-State Management Service

Peng Sun
Ratul Mahajan, Jennifer Rexford,
Lihua Yuan, Ming Zhang, Ahsan Arefin
Princeton & Microsoft
Complex Infrastructure
## Complex Infrastructure

### Microsoft Azure

<table>
<thead>
<tr>
<th>Number of</th>
<th>2010</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Center</td>
<td>A few</td>
</tr>
<tr>
<td>Network Device</td>
<td>1,000s</td>
</tr>
<tr>
<td>Network Capacity</td>
<td>10s of Tbps</td>
</tr>
</tbody>
</table>
## Complex Infrastructure

### Microsoft Azure

<table>
<thead>
<tr>
<th>Number of</th>
<th>2010</th>
<th>2014</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Center</td>
<td>A few</td>
<td>10s</td>
</tr>
<tr>
<td>Network Device</td>
<td>1,000s</td>
<td>10s of 1,000s</td>
</tr>
<tr>
<td>Network Capacity</td>
<td>10s of Tbps</td>
<td>Pbps</td>
</tr>
</tbody>
</table>
## Complex Infrastructure

### Microsoft Azure

<table>
<thead>
<tr>
<th>Number of</th>
<th>2010</th>
<th>2014</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Center</td>
<td>A few</td>
<td>10s</td>
</tr>
<tr>
<td>Network Device</td>
<td>1,000s</td>
<td>10s of 1,000s</td>
</tr>
<tr>
<td>Network Capacity</td>
<td>10s of Tbps</td>
<td>Pbps</td>
</tr>
</tbody>
</table>

Variety of vendors/models/time
Management Applications
Management Applications

Traffic Engineering
Management Applications

- Traffic Engineering
- Load Balancing
Management Applications

- Traffic Engineering
- Load Balancing
- Link Corruption Mitigation
Management Applications

- Traffic Engineering
- Load Balancing
- Link Corruption Mitigation
- Device Firmware Upgrade
Our Question

How to safely run multiple management applications on shared infrastructure
Naïve Solution

- Run independently

Diagram:
- Traffic Engineering
- Link Corruption Mitigation
- Firmware Upgrade
- Network Devices
Naïve Solution

- It does not work due to 2 problems

Traffic Engineering

Firmware Upgrade

Link Corruption Mitigation

Network Devices
Problem #1: Conflict
Problem #1: Conflict

Link-corruption-mitigation adjusts traffic away from Core1
Problem #1: Conflict

Core 1 2
Agg A

Link-corruption-mitigation adjusts traffic away from Core1

Agg B

TE tunes traffic among links to Core1, 2
Problem #2: Safety Violation

Core1 2
Agg A
Agg B
ToRs
Problem #2: Safety Violation

Link-corruption-mitigation shuts down faulty Agg A
Problem #2: Safety Violation

Link-corruption-mitigation shuts down faulty Agg A

Firmware-upgrade schedules Agg B to upgrade
Potential Solution #1

- Traffic Engineering
- Link Corruption Mitigation
- Firmware Upgrade
Potential Solution #1

• One monolithic application
Potential Solution #1

• One monolithic application
• Central control of all actions
Too Complex to Build

• Difficult to develop
  • Combine all applications that are already individually complicated
Too Complex to Build

- Difficult to develop
  - Combine all applications that are already individually complicated

- High maintenance cost
  - for such huge software in practice
Potential Solution #2

Traffic Engineering

Link Corruption Mitigation

Firmware Upgrade
Potential Solution #2

- Explicit coordination among applications

Diagram:
- Traffic Engineering
- Link Corruption Mitigation
- Firmware Upgrade
Potential Solution #2

- Explicit coordination among applications
- Consensus over network changes
Still Too Complex

- Hard to understand each other
- Diverse network interactions
### Still Too Complex

- Hard to understand each other
- Diverse network interactions

<table>
<thead>
<tr>
<th>Application</th>
<th>Routing</th>
<th>Device Config</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traffic Engineering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Firmware upgrade</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Still Too Complex

- Hard to understand each other
- Diverse network interactions

<table>
<thead>
<tr>
<th>Application</th>
<th>Routing</th>
<th>Device Config</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traffic Engineering</td>
<td>![Checkmark]</td>
<td>![X]</td>
</tr>
<tr>
<td>Firmware upgrade</td>
<td>![Checkmark]</td>
<td>![X]</td>
</tr>
</tbody>
</table>
### Still Too Complex

- Hard to understand each other
- Diverse network interactions

<table>
<thead>
<tr>
<th>Application</th>
<th>Routing</th>
<th>Device Config</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traffic Engineering</td>
<td>![check]</td>
<td>![x]</td>
</tr>
<tr>
<td>Firmware upgrade</td>
<td>![x]</td>
<td>![check]</td>
</tr>
</tbody>
</table>
Main Enemy: Complexity

• Application development
• Application coordination
Main Enemy: Complexity

• Application development
• Application coordination

- Independent
- Explicitly coordinate
- Monolithic

Simple

Complex
What We Advocate

• Loose coupling of applications
• Design principle:
  • Simplicity with safety guarantees
What We Advocate

• Loose coupling of applications
• Design principle:
  • Simplicity with safety guarantees
• Forgo joint optimization
  • Worthwhile tradeoff for simplicity
  • Applications could do it out-of-band
Overview of Statesman

• Network operating system for safe multi-application operation
Overview of Statesman

- Network operating system for safe multi-application operation
- Uses network state abstraction
  - Three views of network state
Overview of Statesman

• Network operating system for safe multi-application operation

• Uses network state abstraction
  • Three views of network state
  • Dependency model of states
The “State” in Statesman

- Complexity of dealing with devices
  - Heterogeneity
  - Device-specific commands

Network Devices
The “State” in Statesman

- Complexity of dealing with devices
- Heterogeneity
- Device-specific commands
## State Variable Examples

<table>
<thead>
<tr>
<th>State Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device Power Status</td>
<td>Up, down</td>
</tr>
<tr>
<td>Device Firmware</td>
<td>Version number</td>
</tr>
<tr>
<td>Device SDN Agent Boot</td>
<td>Up, down</td>
</tr>
<tr>
<td>Device Routing State</td>
<td>Routing rules</td>
</tr>
<tr>
<td>Link Admin Status</td>
<td>Up, down</td>
</tr>
<tr>
<td>Link Control Plane</td>
<td>BGP, OpenFlow, ...</td>
</tr>
</tbody>
</table>
Simplify Device Interaction

Past

Application

Network Devices

Now

Application

Network State

Network Devices
Simplify Device Interaction

**Past**

- Application
- Device Statistics

SNMP, OF, vendor API, ...

**Now**

- Application
- Network State

Network Devices
Simplify Device Interaction

Past

Application

Device Statistics

SNMP, OF, vendor API, ...

Device-specific cmds

Network Devices

Now

Application

Network State

Network Devices
Simplify Device Interaction

Past

Application

Device Statistics

Device-specific cmds

SNMP, OF, vendor API, ...

Network Devices

Now

Application

Network State

Read

Network Devices
Simplify Device Interaction

Past

Application

Device Statistics

SNMP, OF, vendor API, ...

Device-specific cmds

Network Devices

Now

Application

Network State

Read

Write

Network Devices
Views of Network State
Views of Network State

<table>
<thead>
<tr>
<th>Observed State</th>
<th>Actual state of the whole network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target State</td>
<td>Desired state to be updated on the whole network</td>
</tr>
</tbody>
</table>

Observed State

Target State

Network Devices

Application
Two Views Are Not Enough
## Two Views Are Not Enough

**One More View**

| Proposed State | A group of entity-variable-values desired by an application |

**Network Devices**

- **Observed State**
- **Proposed State**
- **Target State**
Two Views Are Not Enough

One More View

Proposed State

A group of entity-variable-values desired by an application

Observed State

Proposed State

Target State

Network Devices
Two Views Are Not Enough

One More View

Proposed State

A group of entity-variable-values desired by an application

Observed State

Target State

Network Devices
How Merging Works

• Combine multiple proposed states into a safe target state
How Merging Works

• Combine multiple proposed states into a safe target state

• Conflict resolution
  • Last-writer-wins
  • Priority-based locking
  • *Sufficient for current deployment*
How Merging Works

• Combine multiple proposed states into a safe target state

• Conflict resolution
  • Last-writer-wins
  • Priority-based locking
  • Sufficient for current deployment

• Safety invariant checking
  • Partial rejection & Skip update
Choose Safety Invariants
Choose Safety Invariants

Loose ← Tight
Choose Safety Invariants

Hinder application too frequently

Loose ← — Tight
Choose Safety Invariants

Loose

Cannot protect network operation

Tight

Hinder application too frequently
Choose Safety Invariants

- Our current choice
  - Connectivity: Every pair of ToRs in one DC is connected
  - Capacity: 99% of ToR pairs have at least 50% capacity
Recap of Three-View Model

- Simplify network management
Recap of Three-View Model

- Simplify network management

Observed State

Proposed State

Target State

Application
Recap of Three-View Model

- Simplify network management

What we see from the network
Recap of Three-View Model

- Simplify network management

What we see from the network

What we want the network to be

Observed State

Proposed State

Target State

Application
Recap of Three-View Model

- Simplify network management

- Observed State: What we see from the network
- Target State: What we want the network to be
- Proposed State: What can be actually done on the network

Statesman

Application
Yet Another Problem

• What’s in Proposed State
  • Small number of state variables that application cares
Yet Another Problem

- What’s in Proposed State
  - Small number of state variables that application cares

- Implicit conflicts arises
Yet Another Problem

- What’s in Proposed State
  - Small number of state variables that application cares

- Implicit conflicts arises
  - Caused by state dependency
Implicit Conflict
Implicit Conflict
Implicit Conflict

TE writes new value of routing state of B for tunneling traffic
Implicit Conflict

TE writes new value of routing state of B for tunneling traffic

Firmware-upgrade writes new value of firmware state of B
Dependency Relations

Device

Link
Dependency Relations

PowerState

Device

Link
Dependency Relations

Device

FirmwareVersion

PowerState

Link
Dependency Relations

Device

ConfigurationState
FirmwareVersion
PowerState

Link
Dependency Relations

Device

- bgpd
- SDN
- FirmwareVersion
- PowerState

Link
Dependency Relations

ConfigurationState

FirmwareVersion

PowerState

AdminState

ConfigurationState

Device

Link
Dependency Relations

- RoutingState
- ConfigurationState
- FirmwareVersion
- PowerState
- ConfigurationState
- AdminState

Device

Link
Dependency Relations

PathState

RoutingState

ConfigurationState

FirmwareVersion

PowerState

ConfigurationState

AdminState

Device

Link
Build in Dependency Model

• Statesman calculates it internally

• Only exposes the result for each state variable
  • Whether the variable is controllable
Statesman System

- Observed State
- Proposed State
- Target State
Statesman System

Storage Service

Observed State

Proposed State

Target State
Statesman System

Storage Service

Observed State

Proposed State

Target State

Monitor
Statesman System

Storage Service

- Observed State
- Proposed State
- Target State

Monitor

Checker
Statesman System

Storage Service

- Observed State
- Proposed State
- Target State

- Monitor
- Checker
- Updater
Deployment Overview

• Operational in Microsoft Azure for 12 months
• Cover 10 DCs of 20K devices
Deployment Overview

• Operational in Microsoft Azure for 12 months

• Cover 10 DCs of 20K devices
Production Applications

• 3 diverse applications built
  • Device firmware upgrade
  • Link corruption mitigation
  • Traffic engineering
Production Applications

- 3 diverse applications built
  - Device firmware upgrade
  - Link corruption mitigation
  - Traffic engineering

- Finish within months

- Only thousands of lines of code
Case #1: Resolve Conflict

Inter-DC TE & Firmware-upgrade

DC = Data Center
BR = Border Router
Empty (0%) • Low (1~40%) • Medium (40%~80%) • High (80%~100%)

Time Series in Minutes

Link Index
Empty (0%) • Low (1~40%) • Medium (40%~80%) • High (80%~100%)
TE fails to acquire lock, and moves traffic away
TE fails to acquire lock, and moves traffic away
BR1 firmware upgrade starts...
BR1 firmware upgrade starts

BR1 firmware upgrade ends. Lock released.
BR1 firmware upgrade starts

TE re-acquires lock, and moves traffic back
BR1 firmware upgrade starts

TE re-acquires lock, and moves traffic back
Case #1 Summary

• Each application:
  • Simple logic
  • Unaware of the other

• Statesman enables:
  • Conflict resolution
  • Necessary coordination
Case #2: Maintain Capacity Invariant

Firmware-upgrade & Link-corruption-mitigation

- Link corrupting packets

Diagram showing network connectivity from Core, Agg, and ToR layers to Pod 1, Pod 4, and Pod 10.
Upgrade proceeds in normal speed in Pod 3 and 5
Upgrade proceeds in normal speed in Pod 3 and 5
Upgrade proceeds in normal speed in Pod 3 and 5
Upgrade proceeds in normal speed in Pod 3 and 5.

Upgrade in Pod 4 is slowed down by checker due to lost capacity.
Upgrade proceeds in normal speed in Pod 3 and 5.

Upgrade in Pod 4 is slowed down by checker due to lost capacity.
Case #2 Summary

• Statesman:
  • Automatically adjusts application progresses
  • Keeps the network within safety requirements
Conclusion

- Need network operating system for multiple management applications
Conclusion

• Need network operating system for multiple management applications

• Statesman
  • Loose coupling of applications
  • Network state abstraction
Conclusion

• Need network operating system for multiple management applications

• Statesman
  • Loose coupling of applications
  • Network state abstraction

• Deployed and operational in Azure
Thanks!

Questions?

Check paper for related works