

FROM NETWORK ADMINISTRATOR TO DOMAIN SCIENTIST: CHALLENGES WITH CREATING USABLE HIGH SPEED NETWORKS

SEAN DONOVAN

RUSS CLARK

JERONIMO BEZERRA

GEORGIA TECH
GEORGIA TECH/SOX

FIU

CREATING THE NEXT®

What do domain scientists need to do?

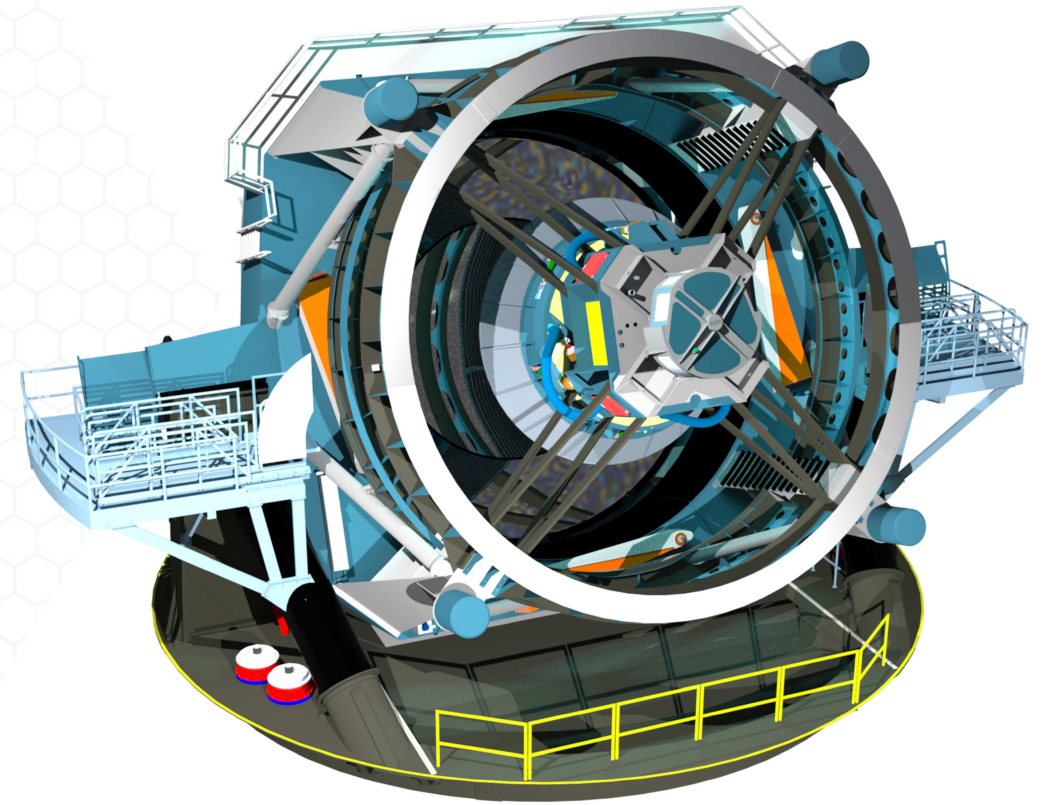
- Move bulk data
- Move streams of data
- Connect instruments with data processing
- Share resources

What do domain scientists know about networks?

- Assume nothing
- Bandwidth (Mbps/Gbps), VLANs, regional networks, research networks may be a foreign concepts

USE CASE: ASTRONOMERS WANT DATA FROM TELESCOPES

- Two possible types of data
 - Bulk data from previous night's viewing
 - Streamed data from instruments
- Need to get data from source to destination
 - Bulk transfers are simple
 - Streamed data should use dedicated path between instrument and processing center



SETTING UP A NETWORK CONNECTION AS DOMAIN SCIENTIST



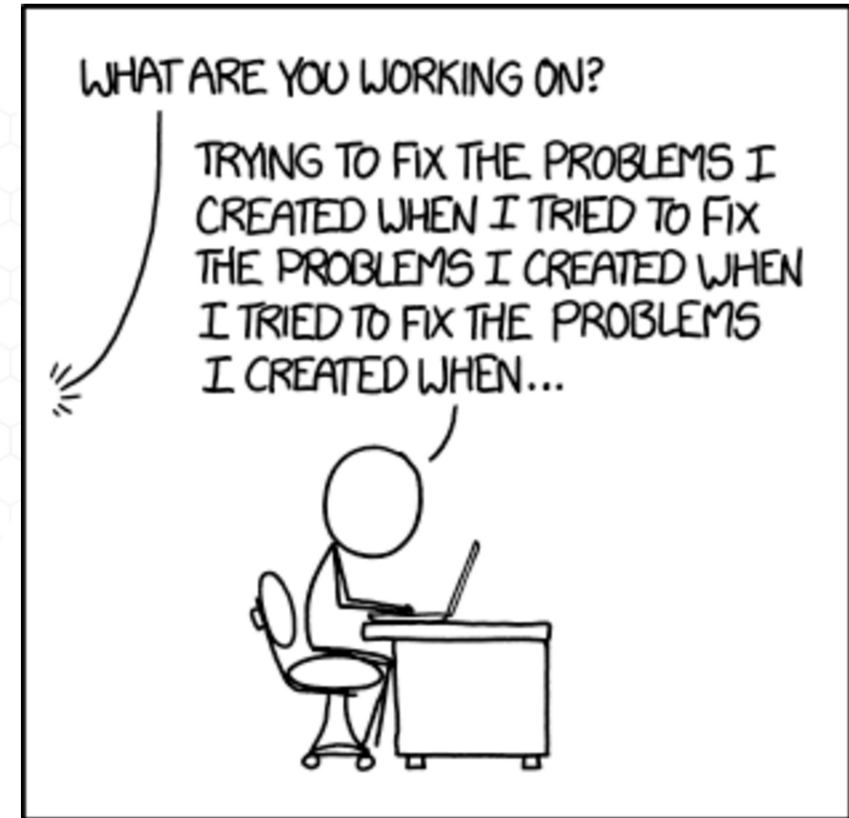
- Call up local IT people
 - Shuffled to the right person, eventually
- Local network admin needs details
 - Bulk transfers? Streaming data? How much? How fast? What timeframes?
- Local network admin needs to talk to destination network admin and network provider(s)
 - To set up path/VLANs/everything else
- Half a dozen or so emails or phone calls
- Days to weeks to set up a connection

- Usual Process

- Setup
- Test
- Fix
- Retest

- Issues

- Slow process – takes days-to-weeks
- Fraught with issues – fat fingering is common
- Too many cooks in the kitchen
- Involves one or two people per network



HOW DO WE IMPROVE THIS?



Method:

- Automated provisioning
- A method for the domain scientists to access the automated system to create own paths

Madness:

- Multi-domain issues abound
- Current process is in place for a number of reasons

WHAT COULD POSSIBLY GO WRONG?



- Domain scientists are not knowledgeable of network terminology
 - Globus is a good example on how to simplify for non-specialists
- Network Operators don't trust users
 - Justifiably!
 - Process is about Approval

A screenshot of the Globus Transfer Files web interface. The top navigation bar is dark blue with the Globus logo and links for "Manage Data", "Publish", "Groups", "Support", and "Account". Below this, a secondary bar contains "Transfer Files" (circled in red), "Activity", "Endpoints", "Bookmarks", and "Console". The main content area is titled "Transfer Files" and features two side-by-side panels, each with an "Endpoint" and "Path" input field and a "Go" button. Below these panels is a "Label This Transfer" section with a text input field. At the bottom, there are "Transfer Settings" with several checkboxes: "sync - only transfer new or changed files", "delete files on destination that do not exist on source", "preserve source file modification times", "verify file integrity after transfer" (checked), and "encrypt transfer". A small link "Get Globus Connect Personal" is at the bottom right.

- Portal for domain scientists
- Simple endpoints
- Submit a request
- Administrators (endpoints and network(s)) approve
 - Automatically or manually
- Integration with transfer applications
 - Globus, GridFTP, etc.



Transfer a File

Users can transfer a file between specified DTNs

Source:

miadtn

Destination:

atldtn

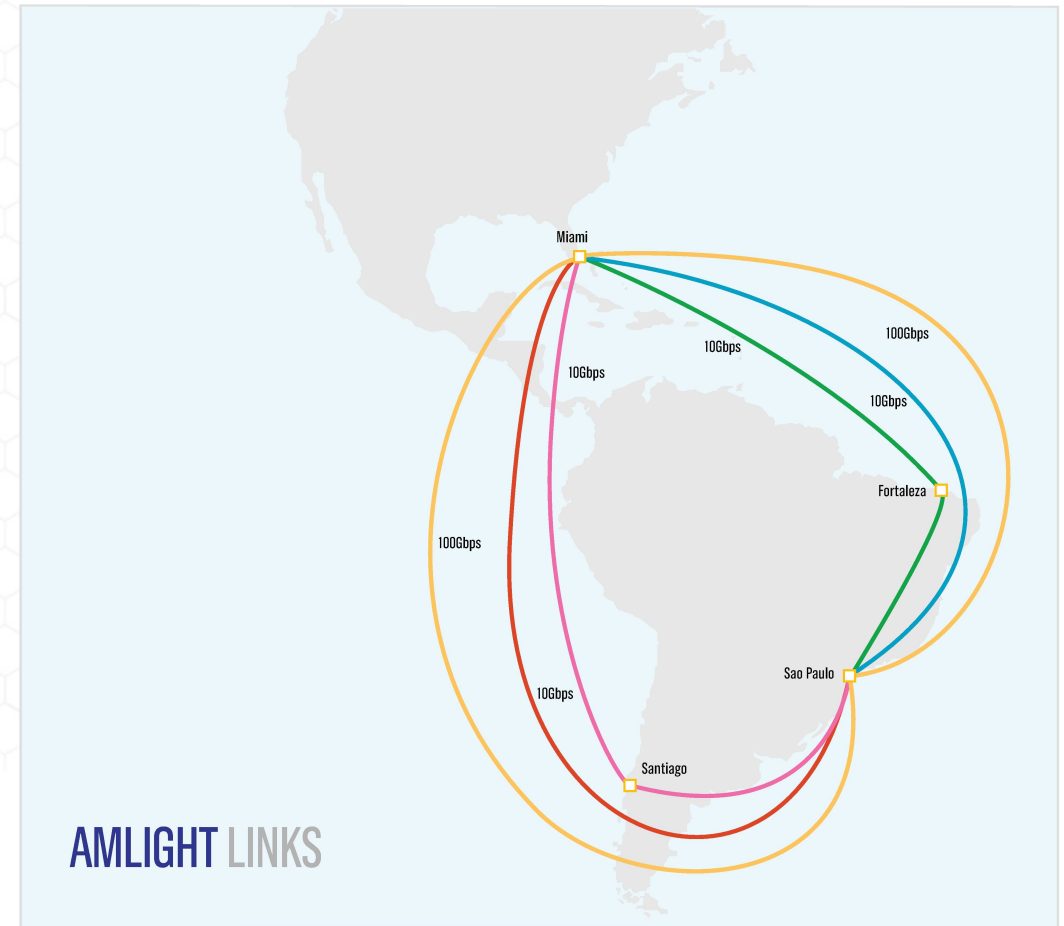
File to Transfer:

sonar-data-2017-01-01.gz

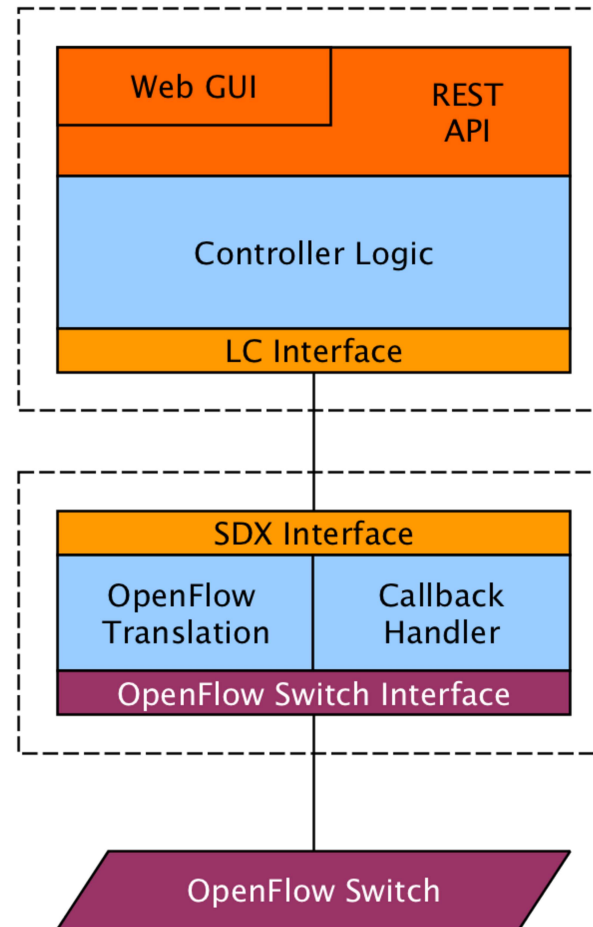
Deadline:

yyyy/mm/dd, --:--

- Inter-domain SDN controller for wide-area, high speed science networks
- We've talked about it before
- Adaptable for multiple situations
 - Including self-service case we're describing
 - Used by not-quite-PhD Joaquin Chung for his thesis



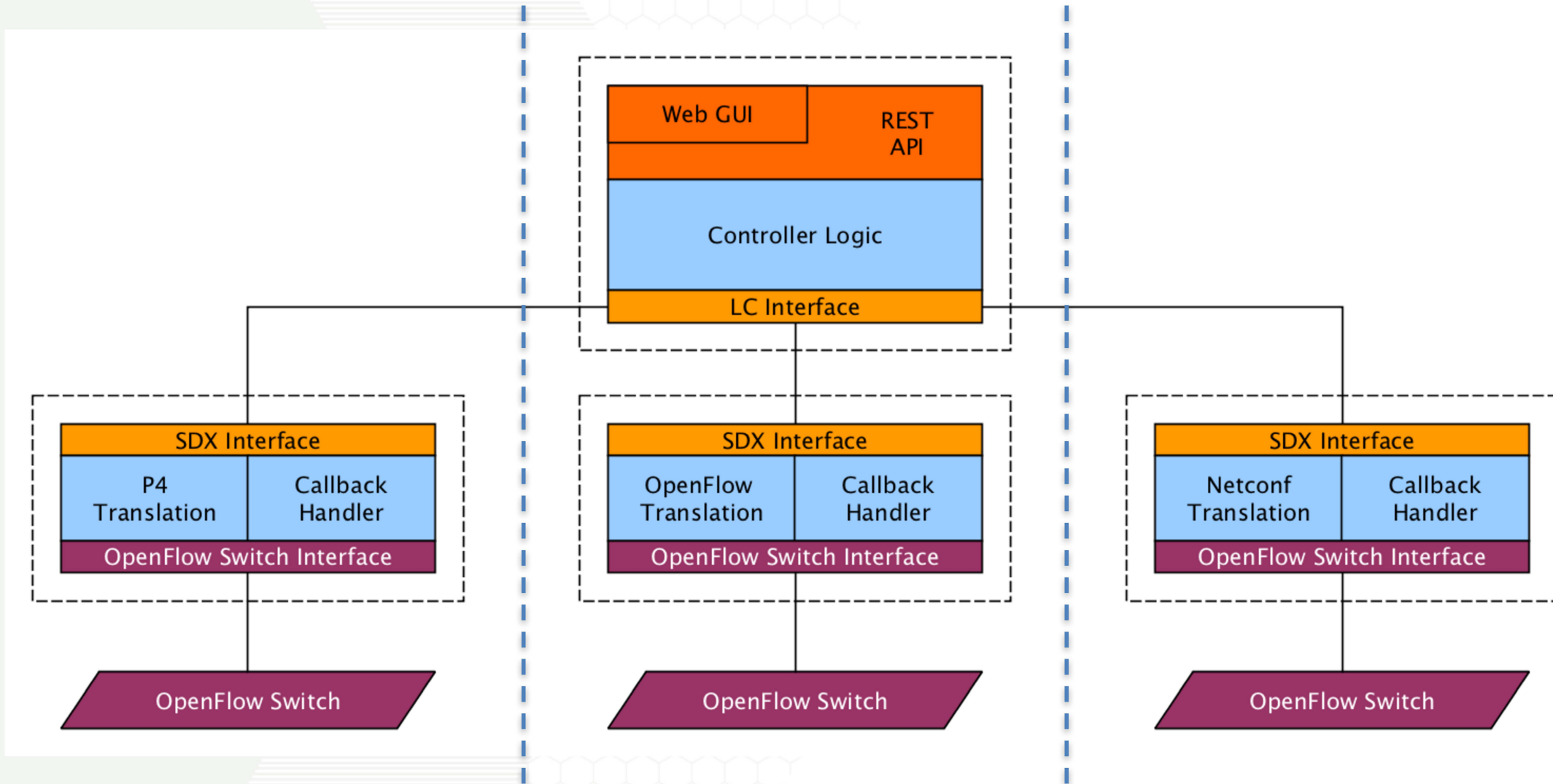
- Astronomer use case
- Network Operator
 - “Advanced options”
- Commercial Operator
 - How to handle per-use costs?
- Multi-domain
 - How to handle different management and authorization levels
- Compute integration
 - Compute resources as a part of the SDX
- What other use cases should we be looking at?
 - Is there a particular type of user that we should be talking to?



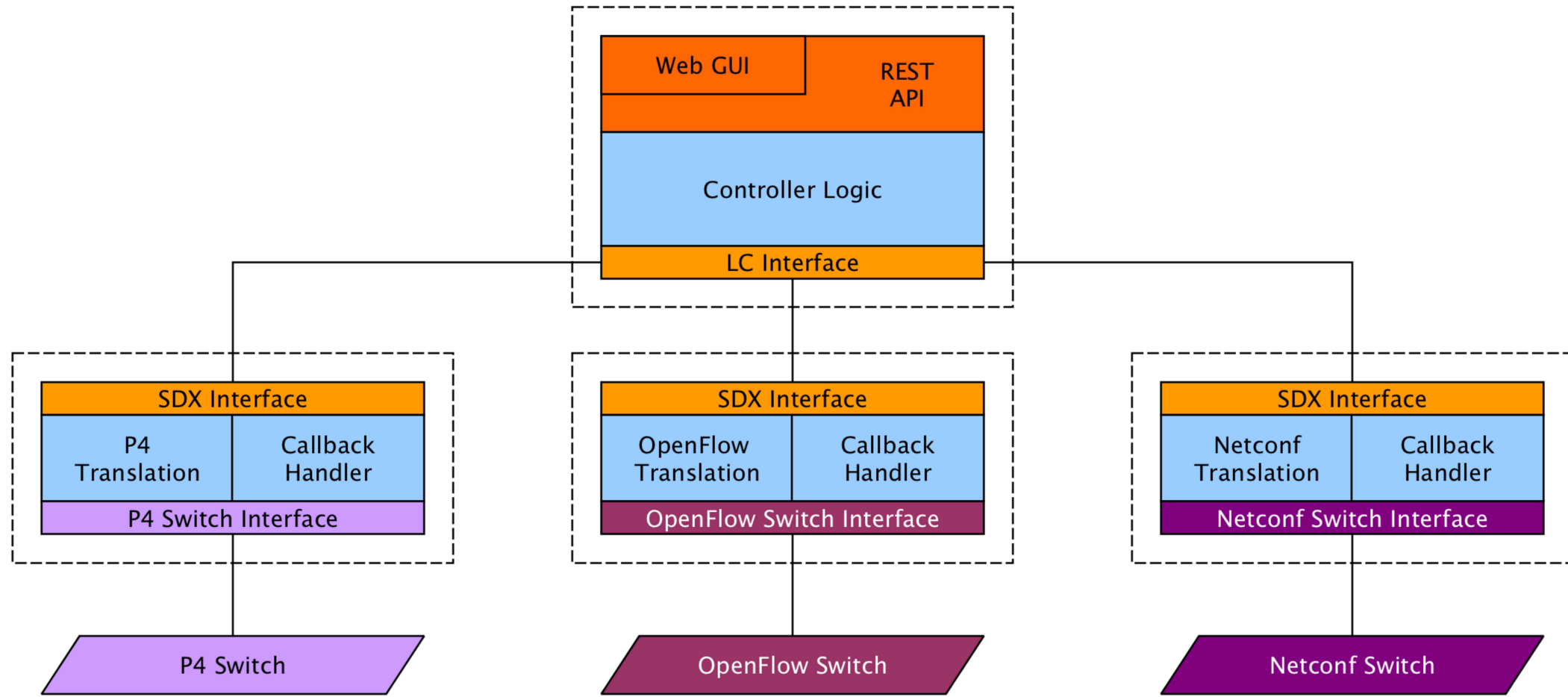
SDX Controller

Local Controller

ATLANTICWAVE/SDX ARCHITECTURE



ATLANTICWAVE/SDX ARCHITECTURE



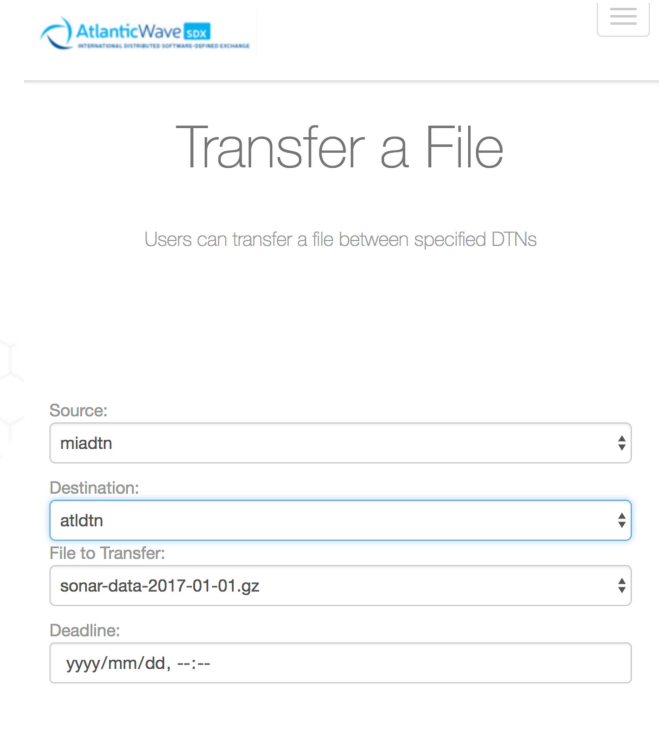
- High Level inputs
 - From users
 - L2 Connection: 2 endpoints, start and stop times, bandwidth required
- Create an L2 tunnel between VLAN 635 on port 7 on ATL switch to VLAN 879 on port 3 on BOS switch between 0000 and 1600 tomorrow with 1gbps reserved
- Mid-level output
 - Not OpenFlow
 - Per-LC rules
- Per LC:
 - ATL: VLAN 635 port 7 to port 12
 - WAS: VLAN 635 port 4 to port 2
 - BAL: VLAN 635 port 8 to port 30
 - NYC: VLAN 635 port 14 to port 24
 - BOS: VLAN 635 port 1 to VLAN 879 port 3

- Mid-level Input
 - E.g., VLAN 635 port 1 to VLAN 879 port 3
- Translates to what switch can handle
 - OpenFlow 1.3 or OF 1.3 + Corsica-specific REST, in our case
 - Could be swapped for some other protocol
 - E.g., Netconf, NSI, P4, Cisco/Juniper/etc. APIs
- Rules
 - `match(inport:1, VLAN:635)`
`action(modify(VLAN=879), fwd(3))`
 - `match(inport:3, VLAN:879)`
`action(modify(VLAN=635), fwd(3))`

HOW DOES THIS ALL FIT TOGETHER?

Scientist wants to transfer a file from
server A to server B across the network

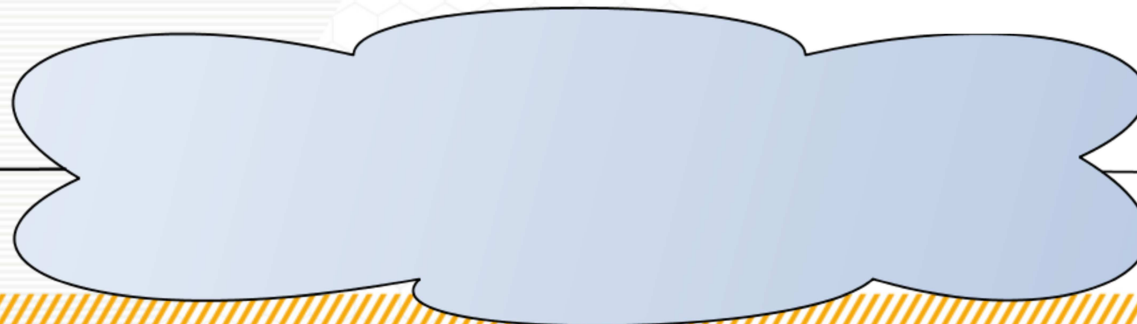
Uses a file transfer interface



The screenshot shows the AtlanticWave file transfer interface. At the top is the AtlanticWave logo with the tagline 'INTERNATIONAL DISTRIBUTED SOFTWARE SERVICES EXCHANGE'. Below the logo is a hamburger menu icon. The main heading is 'Transfer a File'. Underneath is a subtext: 'Users can transfer a file between specified DTNs'. There are four input fields: 'Source:' with a dropdown menu showing 'miadtn', 'Destination:' with a dropdown menu showing 'atldtn', 'File to Transfer:' with a dropdown menu showing 'sonar-data-2017-01-01.gz', and 'Deadline:' with a text input field showing 'yyyy/mm/dd, --:--'.

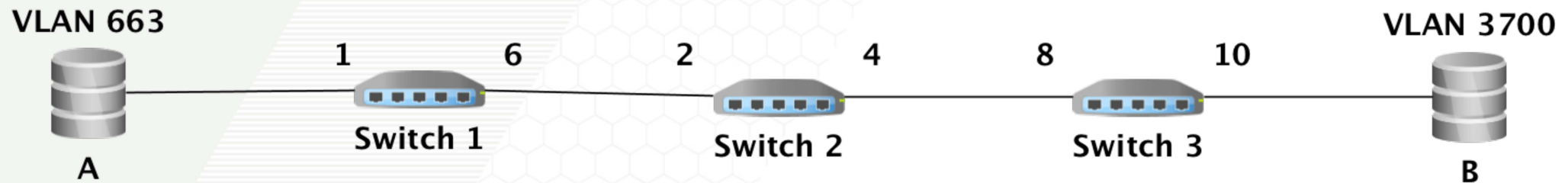


A



B

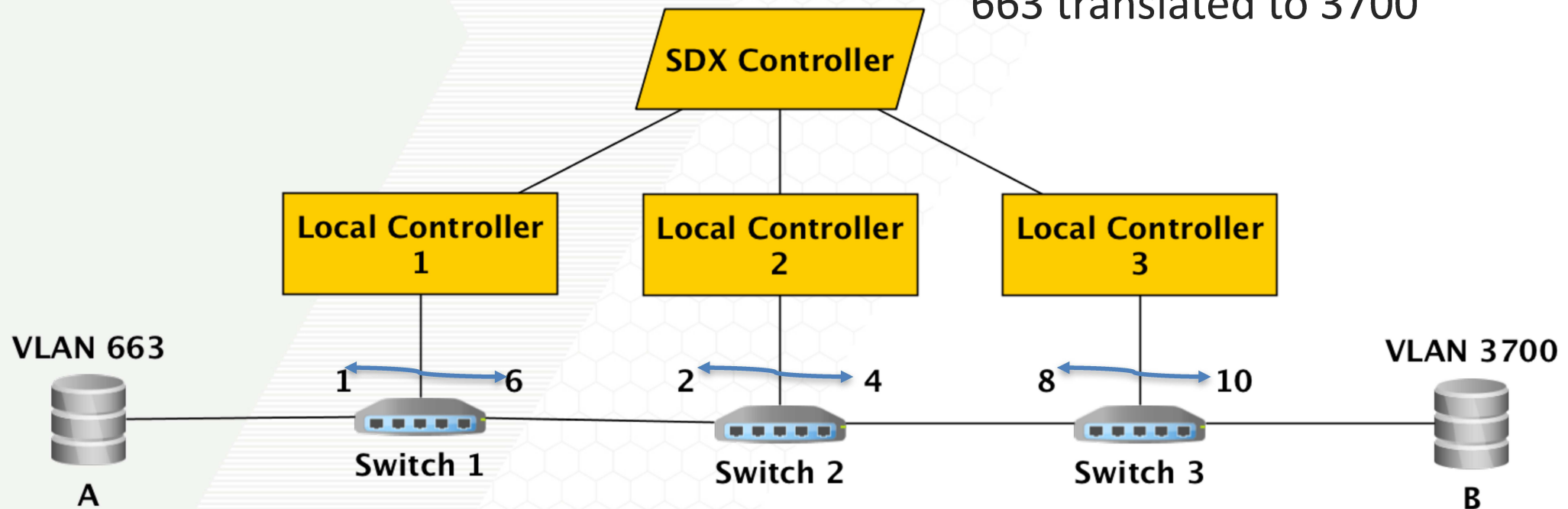
HOW DOES THIS ALL FIT TOGETHER?



HOW DOES THIS ALL FIT TOGETHER?

SDX Controller translates to per-LC rules

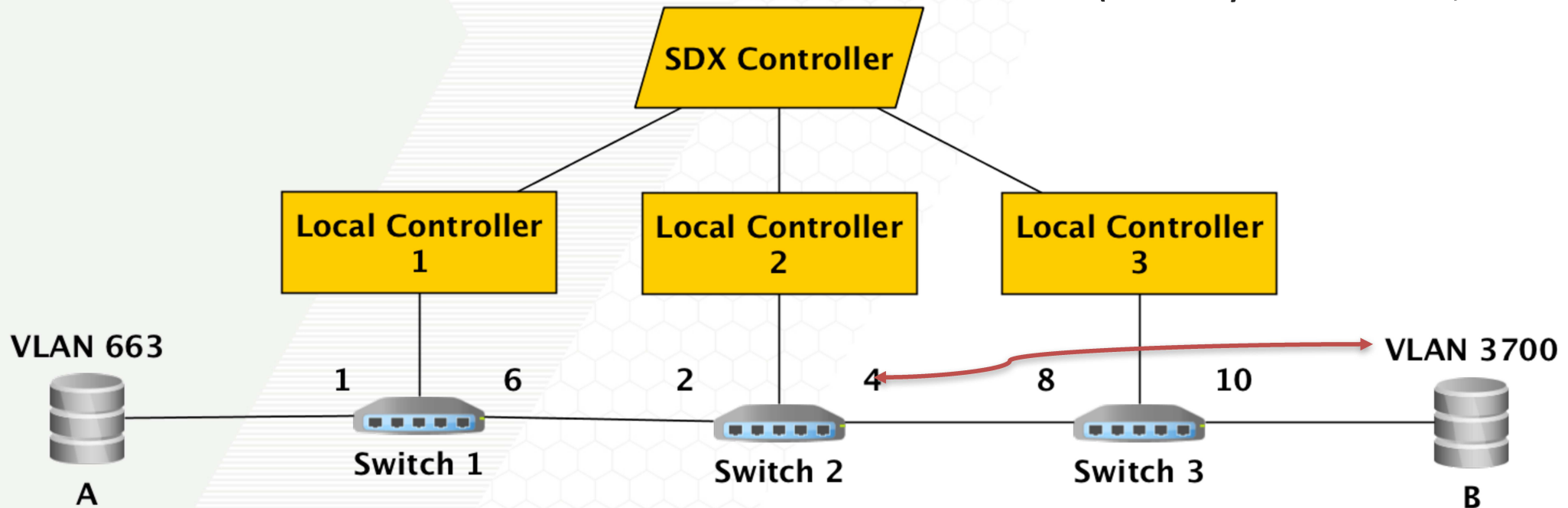
1. Connect ports 1 and 6 on VLAN 663
2. Connect ports 2 and 4 on VLAN 663
3. Connect ports 8 and 10 on VLAN 663 translated to 3700



HOW DOES THIS ALL FIT TOGETHER?

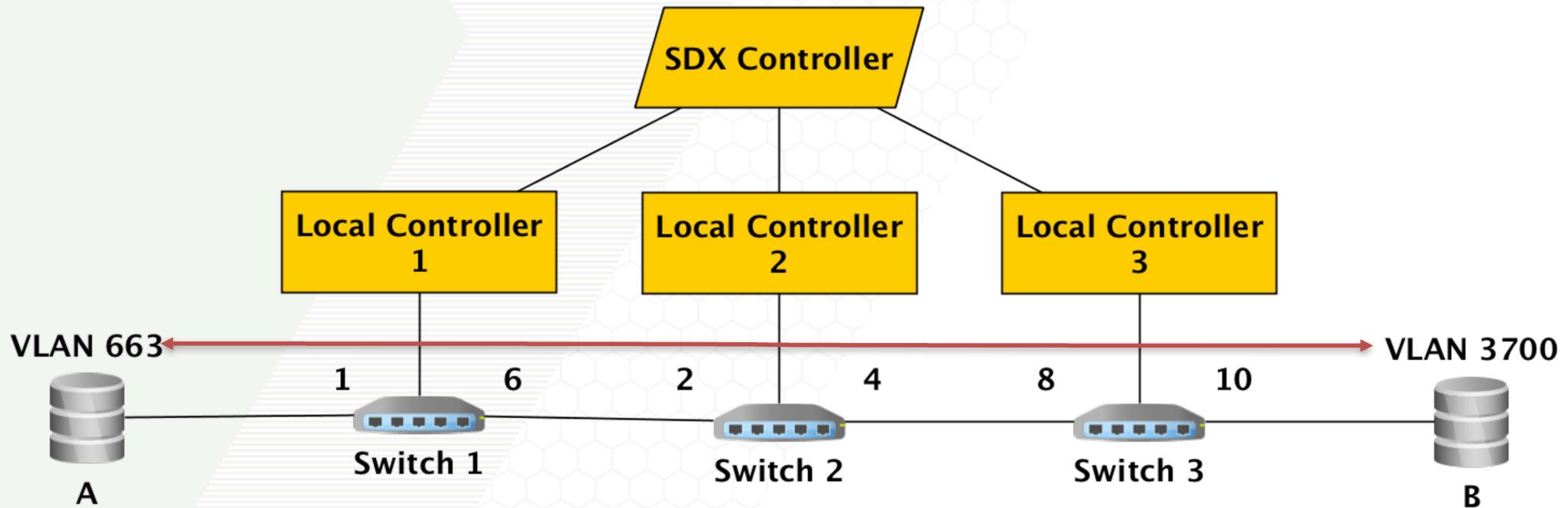
Each Local Controller translates their rules into OpenFlow rules

- match(inport:8, VLAN:663)
action(modify VLAN:3700, fwd:6)
- match(inport:10, VLAN:3700)
action(modify VLAN:663, fwd:4)



HOW DOES THIS ALL FIT TOGETHER?

Each Local Controller translates their rules into OpenFlow rules

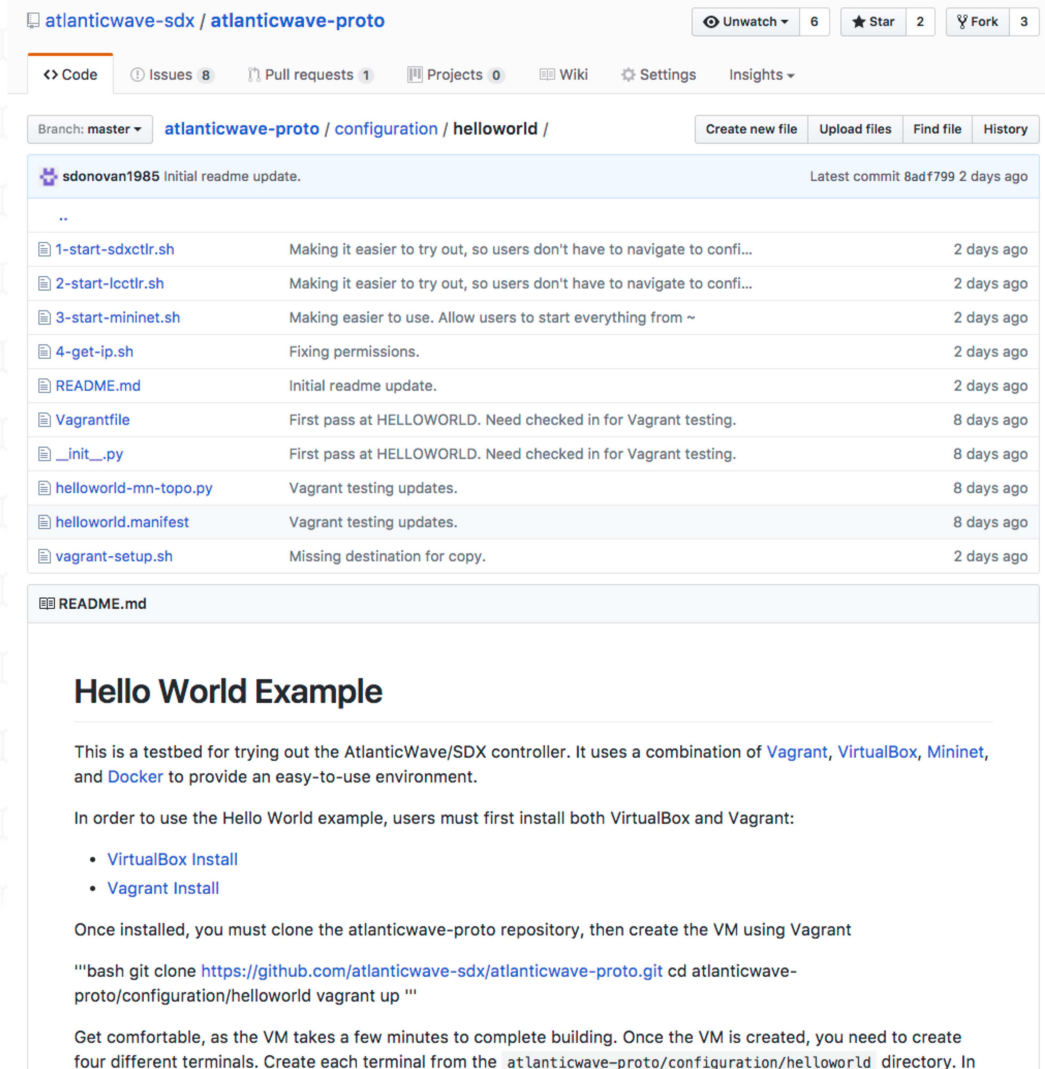


- Multipoint VLANs
- Semi-Arbitrary rules
- REST API
 - For programmatic control

```
{"l2multipoint":{  
  "starttime":"1985-04-12T23:20:50",  
  "endtime":"1985-04-12T23:20:50+0400",  
  "endpoints": [ {"switch":"mia-switch", "port":5, "vlan":286},  
                  {"switch":"atl-switch", "port":3, "vlan":1856},  
                  {"switch":"gru-switch", "port":4, "vlan":3332} ],  
  "bandwidth":1000}}
```

- Integration of compute, not just network
- Due to the uncertainty of big-science funding
 - Where else can this be used?
 - What other use cases are there?
 - Is there a commercial application?
- Pay-per-use science model
 - Complication
 - Can we predict costs?
 - Auctioning resources?

- Web-based GUI in infancy
 - L2 point-to-point is handled easily here
 - Other functionality is REST API only
- Available for use!
 - Easy-to-deploy VM demo using Vagrant
 - Customizable Docker containers for more than “hello world!” functionality
- Happy to have a Skype/Hangouts call with anyone interested in using the AtlanticWave/SDX controller



The screenshot shows the GitHub repository page for `atlanticwave-sdx / atlanticwave-proto`. The repository has 6 Unwatched items, 2 Stars, and 3 Forks. The main branch is `master`. The repository contains several files and folders, including `1-start-sdxcntrl.sh`, `2-start-lcctrl.sh`, `3-start-mininet.sh`, `4-get-ip.sh`, `README.md`, `Vagrantfile`, `_init_.py`, `helloworld-mn-topo.py`, `helloworld.manifest`, and `vagrant-setup.sh`. The `README.md` file is selected, showing the "Hello World Example" section. The text in the README describes a testbed for trying out the AtlanticWave/SDX controller, which uses a combination of Vagrant, VirtualBox, Mininet, and Docker to provide an easy-to-use environment. It also provides instructions on how to install VirtualBox and Vagrant, and how to clone the repository and create the VM using Vagrant.

atlanticwave-sdx / atlanticwave-proto

Unwatch 6 ★ Star 2 🍴 Fork 3

Code Issues 8 Pull requests 1 Projects 0 Wiki Settings Insights

Branch: master atlanticwave-proto / configuration / helloworld / Create new file Upload files Find file History

sdonovan1985 Initial readme update. Latest commit 8adf799 2 days ago

1-start-sdxcntrl.sh	Making it easier to try out, so users don't have to navigate to confi...	2 days ago
2-start-lcctrl.sh	Making it easier to try out, so users don't have to navigate to confi...	2 days ago
3-start-mininet.sh	Making easier to use. Allow users to start everything from ~	2 days ago
4-get-ip.sh	Fixing permissions.	2 days ago
README.md	Initial readme update.	2 days ago
Vagrantfile	First pass at HELLOWORLD. Need checked in for Vagrant testing.	8 days ago
init.py	First pass at HELLOWORLD. Need checked in for Vagrant testing.	8 days ago
helloworld-mn-topo.py	Vagrant testing updates.	8 days ago
helloworld.manifest	Vagrant testing updates.	8 days ago
vagrant-setup.sh	Missing destination for copy.	2 days ago

README.md

Hello World Example

This is a testbed for trying out the AtlanticWave/SDX controller. It uses a combination of [Vagrant](#), [VirtualBox](#), [Mininet](#), and [Docker](#) to provide an easy-to-use environment.

In order to use the Hello World example, users must first install both VirtualBox and Vagrant:

- [VirtualBox Install](#)
- [Vagrant Install](#)

Once installed, you must clone the atlanticwave-proto repository, then create the VM using Vagrant

```
""bash git clone https://github.com/atlanticwave-sdx/atlanticwave-proto.git cd atlanticwave-proto/configuration/helloworld vagrant up ""
```

Get comfortable, as the VM takes a few minutes to complete building. Once the VM is created, you need to create four different terminals. Create each terminal from the `atlanticwave-proto/configuration/helloworld` directory. In

- Domain scientists are network users, but they are not network operators
- They should be able to get network resources easily
- Automation of network resources should follow the self-service model used by compute resources today
- The AtlanticWave/SDX controller is working on being an example network resource allocation mechanism
- The controller design naturally lends itself to multi-domain situations
- It can be used as a prototype to build off of

QUESTIONS?

CREATING THE NEXT®

CONTACT



Sean Donovan

sdonovan@gatech.edu

Russ Clark

russ.clark@gatech.edu

Jeronimo Bezerra

jbezerra@fiu.edu

<http://www.atlanticwave-sdx.net/>
<https://github.com/atlanticwave-sdx/>

Supported by
NSF Award ACI-1451024
International Research Network Connections Program
IRNC-RXP

BACKUP

JOAQUIN CHUNG'S THESIS WORK

