Virtual Environment for Testing Software-Defined Networking Solutions for Scientific Workflows

Qiang Liu, Nagi Rao, Satya Sen,
Oak Ridge National Laboratory

Brad Settlemyer, H-B Chen,
Los Alamos National Laboratory

Josh Boley, Raj Kettimuthu,
Argonne National Laboratory

Dimitri Katramatos,
Brookhaven National Laboratory

The picture can't be displayed.

AI-Science 2018 Workshop [in conjunction with HPDC 2018]

June 11, 2018

ORNL is managed by UT-Battelle
for the US Department of Energy

The picture can't be displayed.

Outline

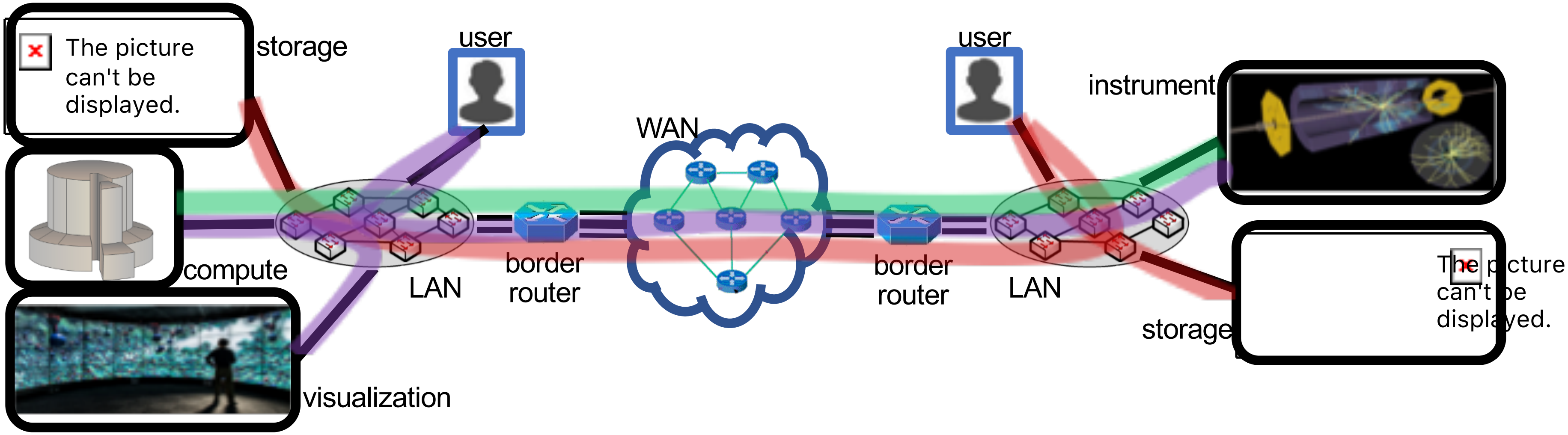
- Introduction
  - Motivation
  - Scientific Workflows
  - Challenges
  - Our Approach
- Virtual Science Network Environment
  - Site-Service Daemon SDN Framework
- Use Cases
  - Lustre File Transfer
  - Streaming Applications
  - Instrument Monitoring and Steering
- Conclusions

Introduction

Introduction

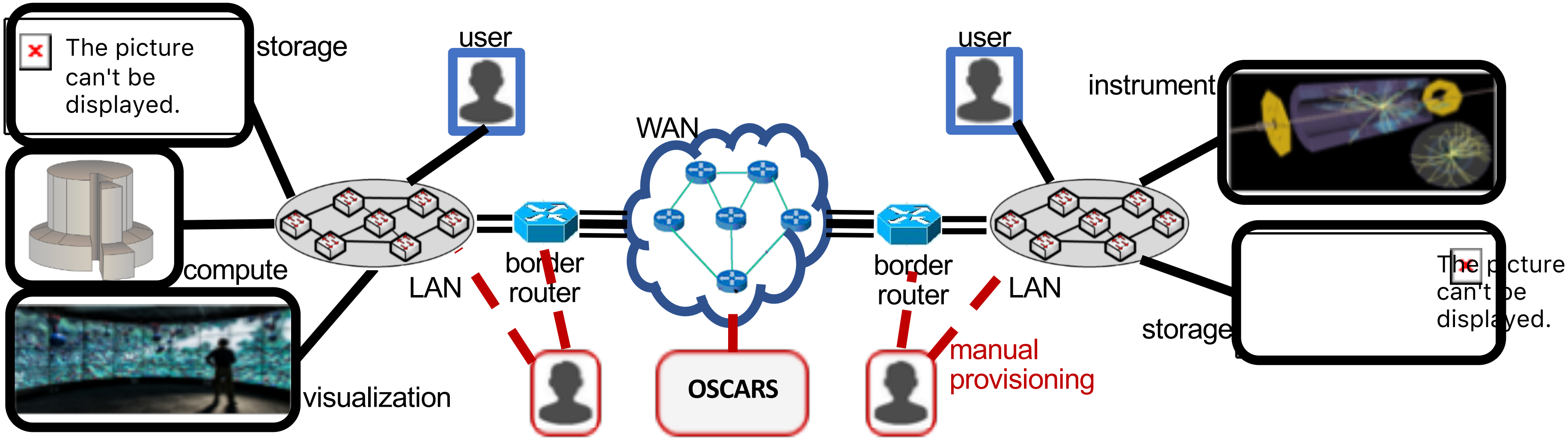
- Goal: Develop the Virtual Science Network Environment (VSNE) for testing software-defined-networking (SDN) solutions for scientific workflows.
- Motivations:
  - Scientific network flows
    - support successful collaborative research in many science areas
      - including biology, climate science, computing, material science, nuclear science, and others;
    - involve a wide variety of tasks
      - using leading-edge supercomputers for high-performance computations and simulations;
      - utilizing large science instruments for conducting experiments;
      - accessing storage systems for retrieving and archiving experimental/simulated data;
    - operate over an increasingly distributed infrastructure with
      - supercomputers, instruments, and storage systems;
      - connected via wide, local, storage area networks.
  - SDN and virtualization technologies have recently shown great promises in supporting fast and robust network capabilities entirely by software
  - VSNE enables early development and testing of SDN scripts and solutions without involving any production-grade physical infrastructure and multi-site collaboration

Scientific Workflows



- Scientific workflows are realized by composing and automating complex applications, while masking the complexity of execution infrastructure
  - ***Workflows for memory and file transfers***
    - Example: In climate science, Earth System Grid Federation (ESGF) grants remote access of the observational/simulated data stored in various distributed data repositories to thousands of users;
  - ***Workflows for near-real-time computations***
    - Example: In cosmology, the data generated by the Palomar Transient Factory (PTF) survey were processed at NERSC/LBNL to identify optical transients within minutes of images being taken;
  - ***Workflows for dynamic monitoring and control***
    - Example: Data generated at various science facilities (e.g., ALS, SNS) are often dynamically monitored to understand whether the simulation/experiment is functioning properly or not.

Challenges



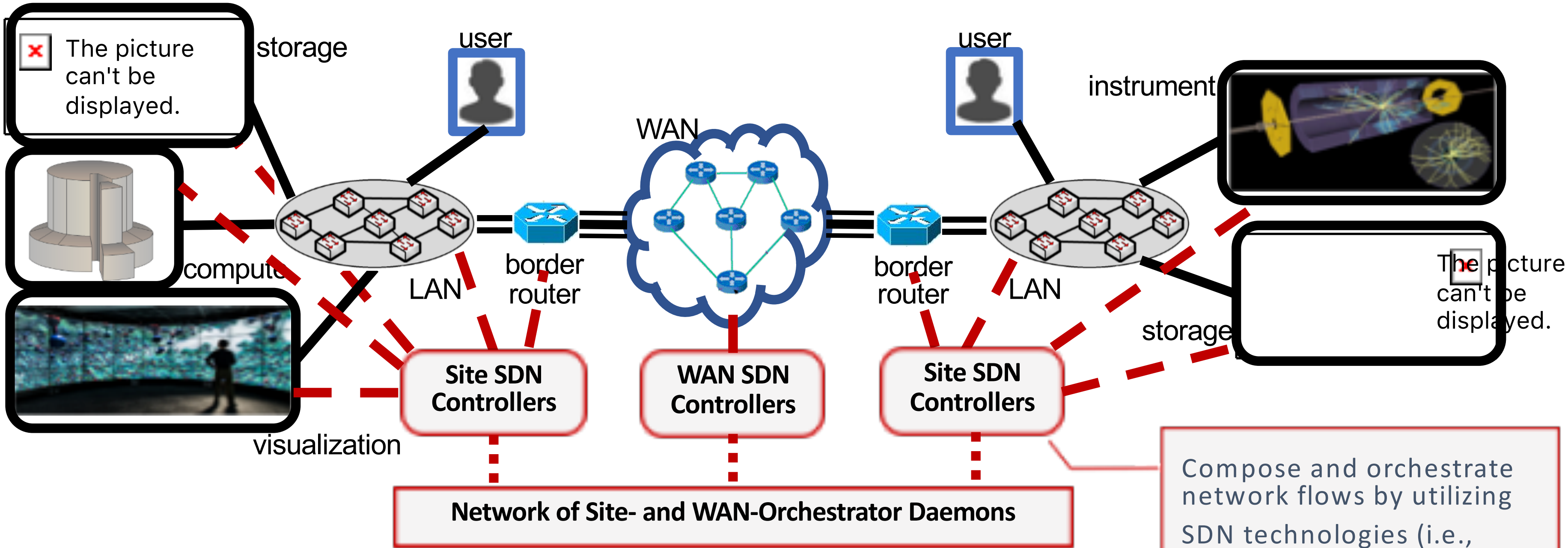
- Custom-designed workflows are composed and configured by teams of experts
  - Takes days, sometimes weeks, to establish an end-to-end dedicated path
  - Number of possible combinations of parameters that need to be optimized to design a complex science flow is increasing exponentially

  → manual composition of optimal flows will soon be impossible to perform
- Individual facilities are only locally optimized

  → lead to misalignments between different subsystems, for example, when a supercomputer is allocated while the network is unavailable
- Multiple resources are over-provisioned to meet the peak transient needs

  → infrastructure is not utilized optimally, such expenses are not justifiable

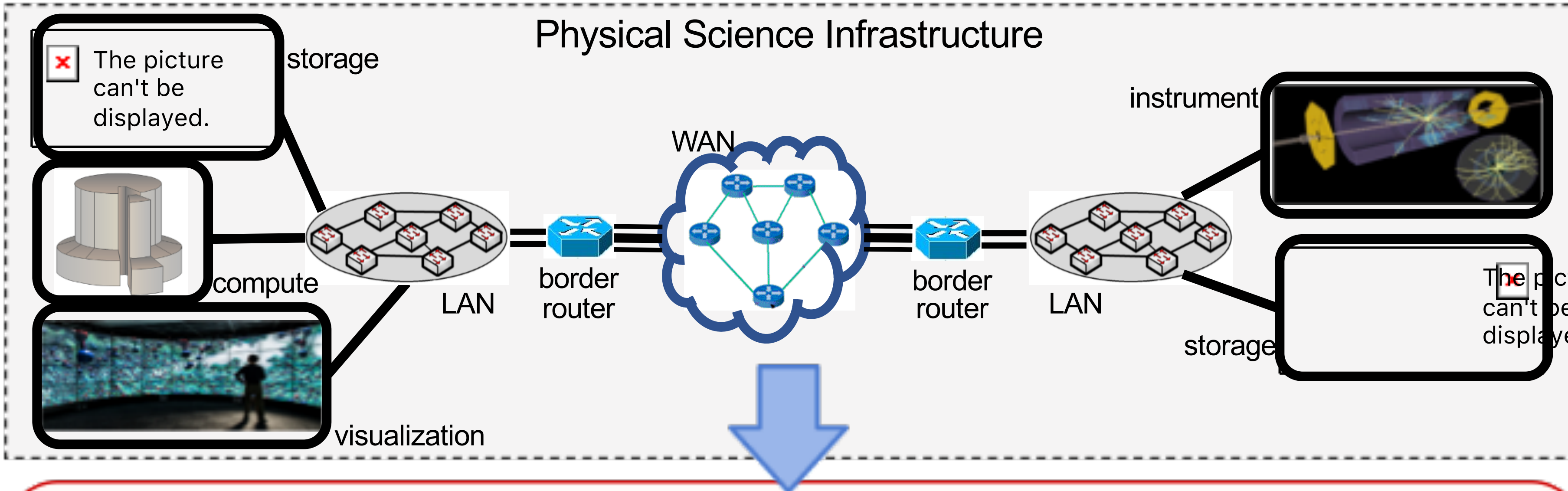
SDN Approach



Compose and orchestrate network flows by utilizing SDN technologies (i.e., without any manual supervision)

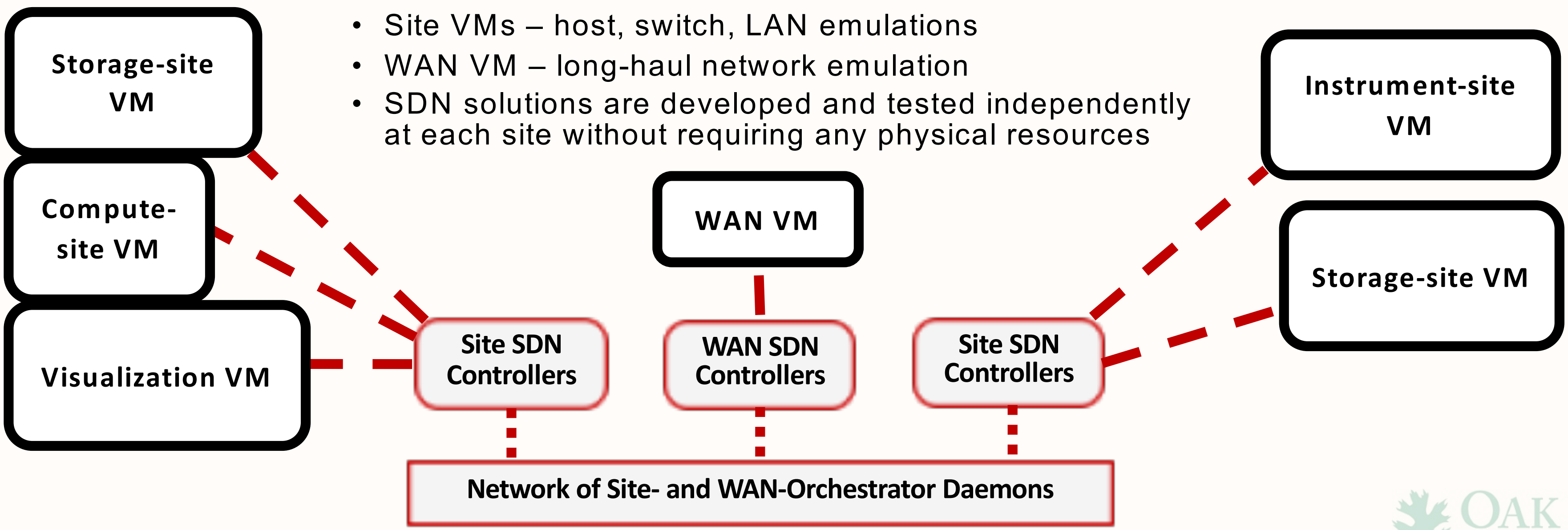
- **Challenges**:
  - labor-intensive coordination among site and network operations teams
  - substantial resource allocations, while taken them away from normal production use
  - disruptive characteristics of SDN codes in the early developmental stages

Our Approach: Development of VSNE



VSNE: emulates sites and network infrastructure

- Site VMs – host, switch, LAN emulations
- WAN VM – long-haul network emulation
- SDN solutions are developed and tested independently at each site without requiring any physical resources



Virtual Science Network Environment

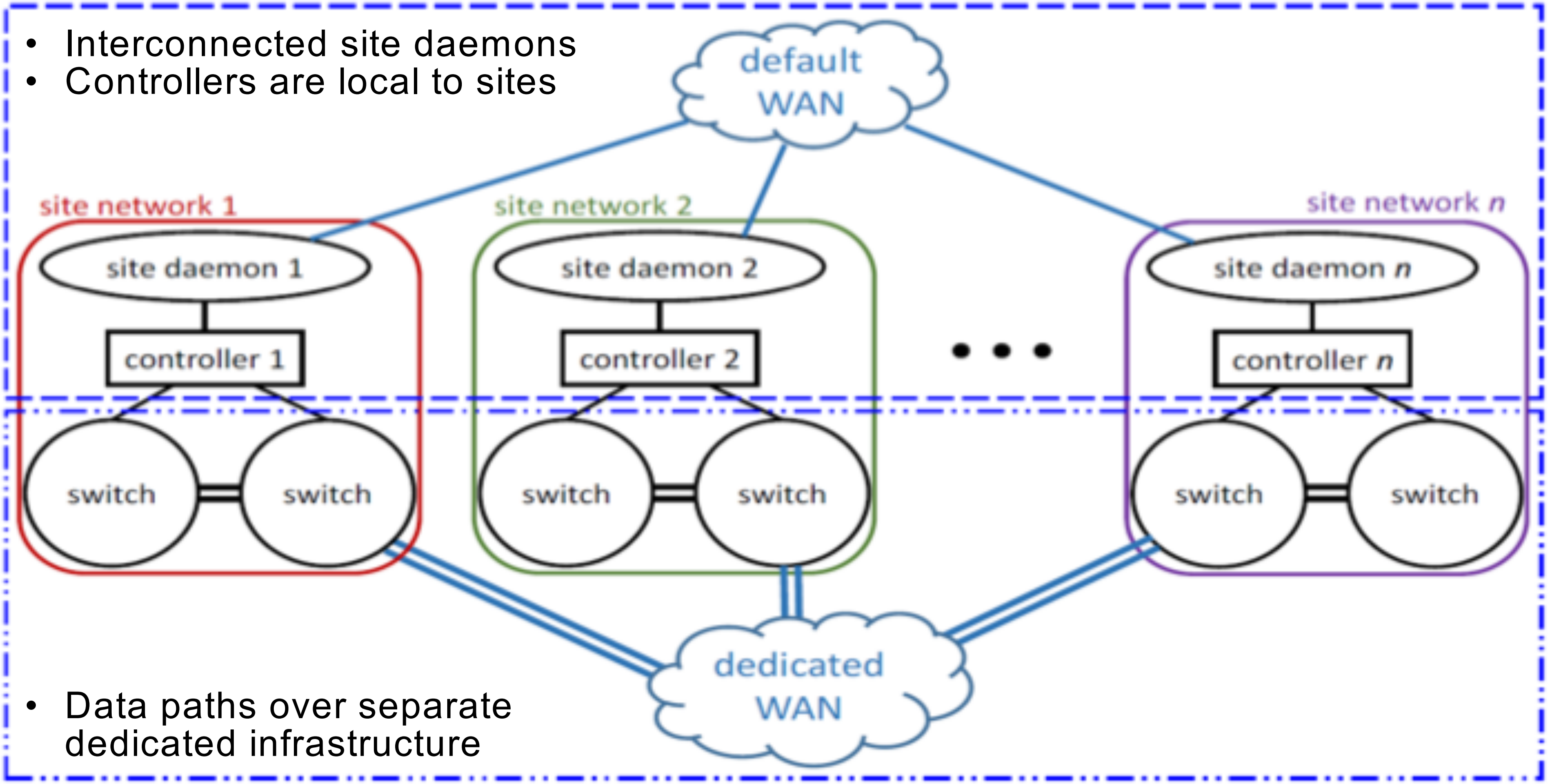
VSNE Overview

- VSNE emulates site and network infrastructure using Virtual Machines (VM)
  - includes custom parameterized topologies with end-hosts, switches, and network links emulated using Mininet
  - emulates science instruments by including corresponding network specifications (switches, links, and end-hosts)
  - supports parallel filesystems (e.g., Lustre) for storage purpose
- SDN controllers (e.g., OpenDaylight, Floodlight, ONOS) orchestrate the network flows
  - by adding/modifying/deleting flows on appropriate OpenFlow-enabled switches
- Site-service daemon framework: A set of site-service daemons maintain persistent connectivity among the VMs, and also with the local site-controllers, switches, and users
- Specifically, the developed VSNE emulates the infrastructure that is currently being built to span ANL, BNL, LANL, and ORNL
  - codes are being developed and tested at each site while
    - hardware is being installed and tested
    - security plans are being developed and approved
  - codes are directly transferable to site hardware upon maturity

OAK RIDGE National Laboratory

Site-Service Daemon Framework

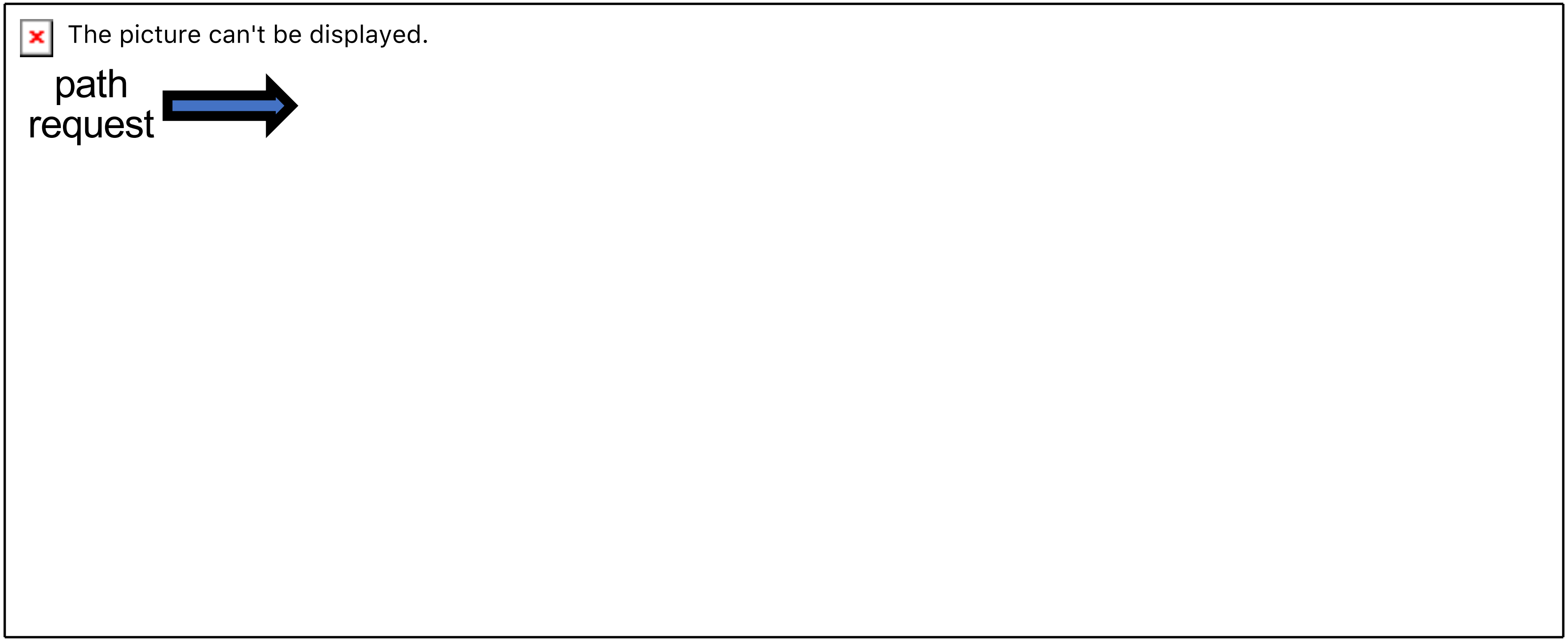
- Interconnected site daemons
- Controllers are local to sites



- Data paths over separate dedicated infrastructure

Site-Service Daemon Framework (contd.)

- **Data paths are setup as needed over dedicated network**:
  - Path requests: User/applications communicate with daemons
  - Path setup: Local daemons receives requests
    - sends requests to remote- and WAN-daemons
    - sets up (or tears down) its local site paths



Oak Ridge National Laboratory

Site-VM Emulation

- Site hosts and switches are emulated using Mininet
- Site-service daemon codes run under linux
- Application codes (for file transfer, streaming, and experiment steering) run under linux, and are made available to all emulated site hosts

The picture can't be displayed.

- Lustre filesystem is supported for storage
  - A Lustre server is run on a dedicated VM
  - Lustre clients are run by site hosts to mount the filesystem
  - Lustre-VM is connected to site-VMs over an internal network that represents the site storage network
- Three network interfaces are enabled
  - NAT interface to connect the guest VM with the host OS
  - One internal interface to enable the control-plane communications
  - Another internal interface to allow the dedicated data-plane connections

WAN-VM Emulation

- WAN switches are created in Mininet to emulate the physical circuits (e.g., OSCARS)
- WAN service daemon codes run under linux
- Long-haul link latency between sites are incorporated in Mininet by imposing various delay parameters
- WAN-VM is being particularly used for code development
  - will be replaced by OSCARS API in deployment

The picture can't be displayed.

The picture can't be displa

RTT

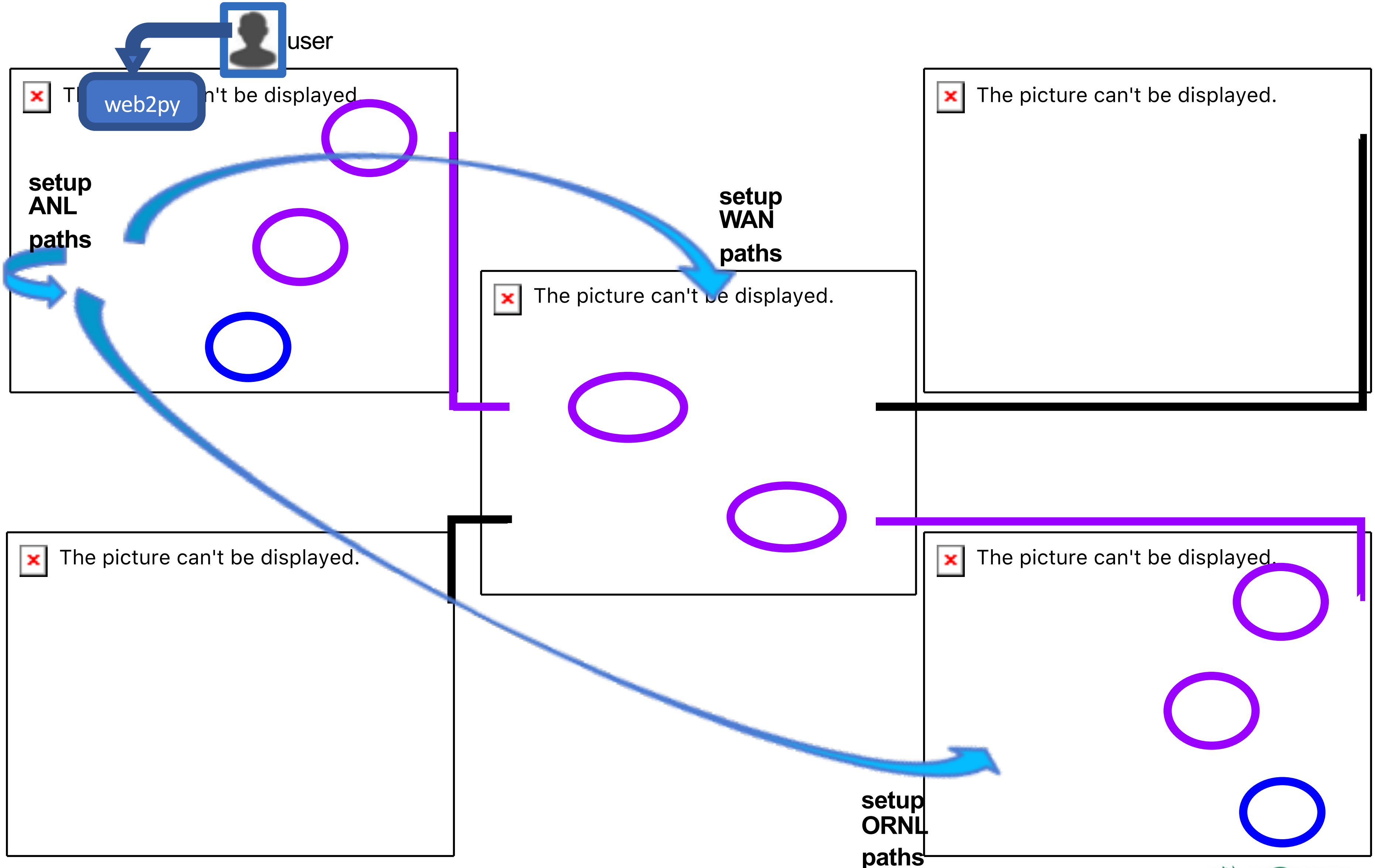
User Web Interface

- A web2py-based web service stack is
  - developed to handle an end-user’s datapath reservation request
  - integrated tightly with the site-service daemon framework so that user’s web-request is redirected to the daemons as CLI requests

The picture can't be displayed.

OAK RIDGE National Laboratory

End-to-end Dedicated Path Setup



OAK RIDGE National Laboratory

Use Case: Lustre File Transfer

Oak Ridge National Laboratory

Use Case: Lustre File Transfer

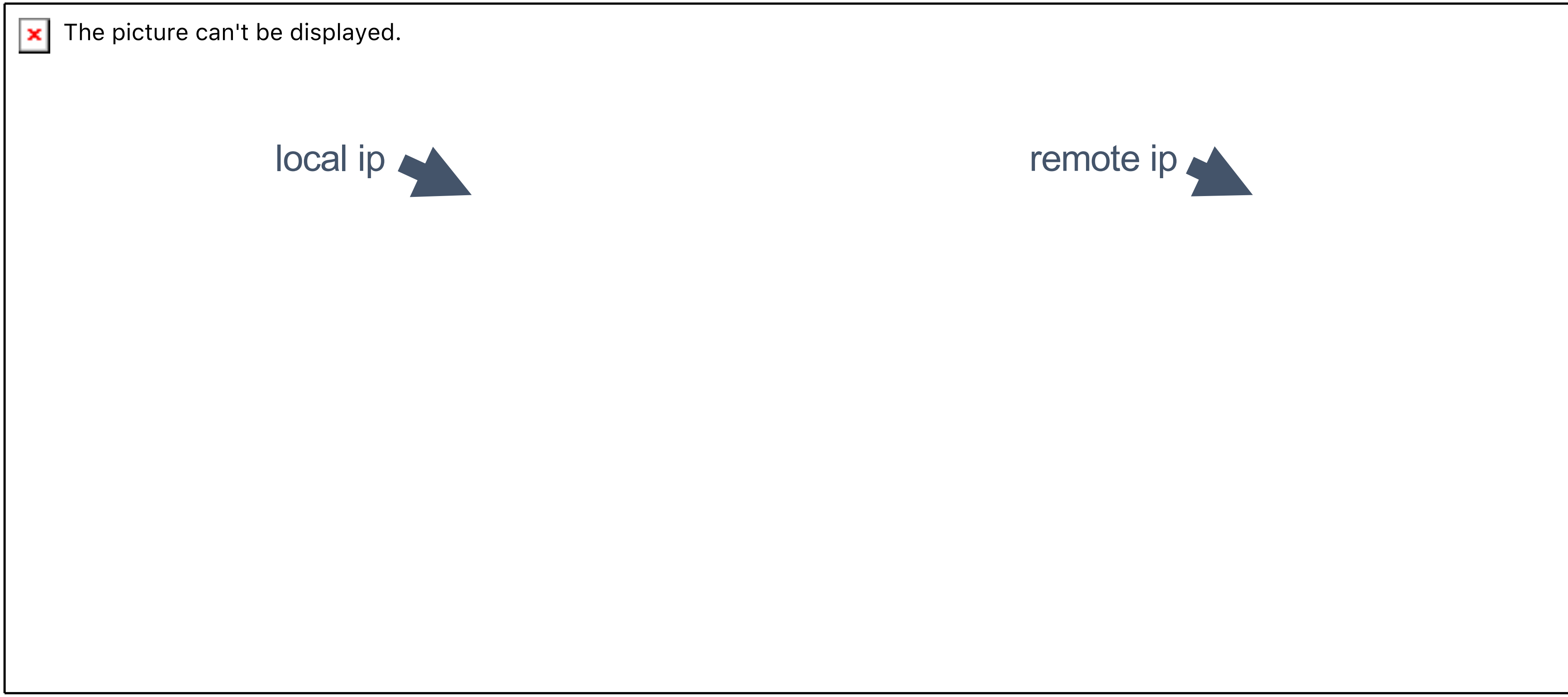
- **Motivation**: Data transfer among various collaborating (remote) sites is one of the most basic tasks for scientific workflows
- **Use case**: Utilize XDD file-transfer tool over the Lustre filesystem to transfer a 50 MB file from h1-anl to h2-bnl
- **Procedure**:
  - Setup a dedicated datapath between the host pair (h1-anl, h2-bnl)
  - Install XDD on the respective site VMs (ANL-VM and BNL-VM)
  - Start the XDD process (XDD-write) on h2-bnl (intended destination of the file)
  - Initiate the XDD process (XDD-read) on h1-anl (sender of the file)
  - Verify the transfer performance at both the sender and receiver

Use Case: Streaming Applications

Use Case: Streaming Applications

- **Motivation**: Virtual collaborative meeting to stream data (image/video) and exchange messages is critical for science applications involving near real-time analysis and data-visualization
- **Red5 streaming framework**:
  - Supports various streaming applications, e.g., chat, video, data streaming
  - Can be easily customized for science collaboration
    - transforming the chat interface into a simple computational monitoring app by posting the outputs from a running computation task using curl scripts
- **Use case**: Use Red5 streaming framework to stream a media file stored on ORNL-VM to both ORNL and BNL hosts
- **Procedure**:
  - Install Red5 server on ORNL-VM
  - Initiate Red5 server from ORNL host h2-ornl, which has access to the media file required for streaming
  - Setup a dedicated data-plane connection from a remote host (h1-bnl) to the Red5 server (h2-ornl)
  - Bring up the streaming app at h1-bnl to visualize the media file

Use Case: Streaming Applications (cont.)



ORNL VM

BNL VM

Use Case: Instrument Monitoring and Steering

Use Case: Instrument Monitoring and Steering

- **Motivation**: Dynamic monitoring of the intermediate results from an instrument and sending control commands to steer the next configurations are extremely critical workflows associated with the scientific instruments

- **Use case**: Emulate an instrument on ANL-VM whose (emulated) output is monitored by a remote host h2-ornl, which in turn sends back some (emulated) monitoring commands to the instrument

- **Procedure**:
  - Emulate an instrument by extending the Mininet environment on ANL-VM
  - Setup a dedicated path for remotely monitoring the instrument from h2-ornl
  - Emulate instrument data flow by running TCP iperf on the instrument host (iperf-client) and monitoring host h2-ornl (iperf-server)
    - data flows for a specific amount of time (‘*intrv*’) in each transfer
    - data transfers are broken (as desired) non-contiguously by periods of no traffic
  - Send emulated control commands from h2-ornl to change the transfer time (‘*intrv*’) to a different value
    - utilize the default IP network (control-plane) to send the control commands

The picture can't be displayed.

Use Case: Instrument Monitoring and Steering

The picture can't be displayed.

ANL-VM

The picture can't be displayed.

‘*intrv*’

The picture can't be displayed.

ORNL-VM

The picture can't be displayed.

modified value of ‘*intrv*’

Conclusions

Conclusions

- We developed a Virtual Science Network Environment (VSNE) for early testing of SDN functionalities for multi-site scientific workflow applications
- VSNE framework
  - utilizes virtual machines, Mininet topologies, custom scripts SDN controllers, and site-service daemons to coordinate among multiple sites
  - does not require immediate deployment or allocation of physical resources
- Demonstrate the viability of the VSNE with three use cases
  - Lustre file transfer
  - Streaming applications
  - Instrument monitoring and steering
- **Future directions:**
  - Incorporation of VSNE into a physical testbed with actual hosts, OpenFlow switches, and long-haul links
    - would provide comparative performance analysis between the physical and virtual environments
  - Development of other proof-of-the-principle functionality tests

Acknowledgements

The picture can't be displayed.

OAK RIDGE
National Laboratory

This work was supported by the High-Performance Networking Program, Office of Advanced Computing Research, U.S. Department of Energy, and by Extreme Scale Systems Center, sponsored by U.S. Department of Defense.

OAK RIDGE
National Laboratory

Questions?