# A Software Defined Network Design for Analyzing Streaming Data in Transit

Ying Liu, Dimitrios Katramatos

SNTA 2019, Phoenix, Arizona

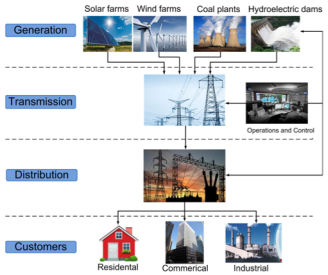
BROOKHAVEN NATIONAL LABORATORY

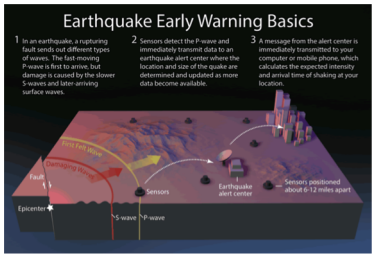
U.S. DEPARTMENT OF ENERGY

# Motivation

- Network traffic keeps reaching new highs as unprecedented volumes of data flow between an **ever-increasing** number of sources and destinations
- We **aim** to develop a **network-centric** approach for streaming data processing to **facilitate** scientific data analysis and **reduce** the **overhead** in sending big data to a data center

# Use cases


- **I.e.**
  - Cybersecurity
  - Smart power grid
  - Weather forecast
  - Intelligent transportation system
  - Earthquake Warning system
- **Characteristics**
  - Time sensitive applications
  - Large-scale (geographically)
  - Time-series data collected from different sources, i.e. sensors, cameras, surveillance video, cellphones, GPS, satellites, etc.
    - Streaming data
    - Burst data (streaming, in short time period)

# Related works

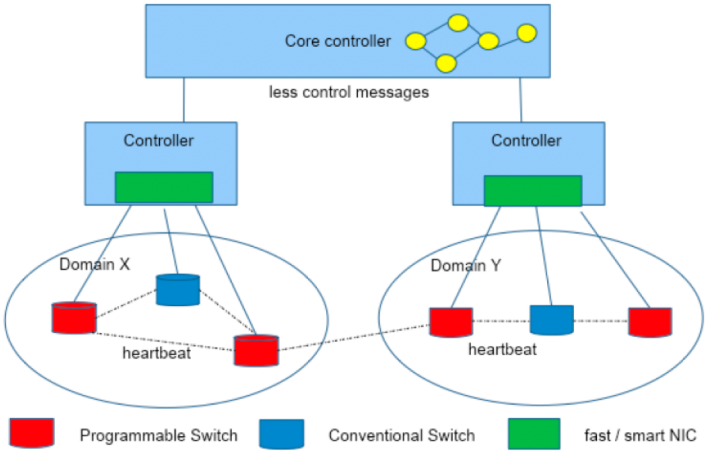
- Current applications of streaming analysis in industry:
  - In servers after network stack
    - Using http / https, FTP, ssh protocols to transmit data and call streaming library (software), i.e. Apache Spark Streaming, Kinesis (Amazon), IBM Streams
  - In server's NIC before network stack
    - AccelNet (Microsoft)
  - Between servers in a data center
    - Catapult (Microsoft), SHArP (Mellanox)
- However, streaming processing in data centers remain less prevalent when compared to offline learning algorithms

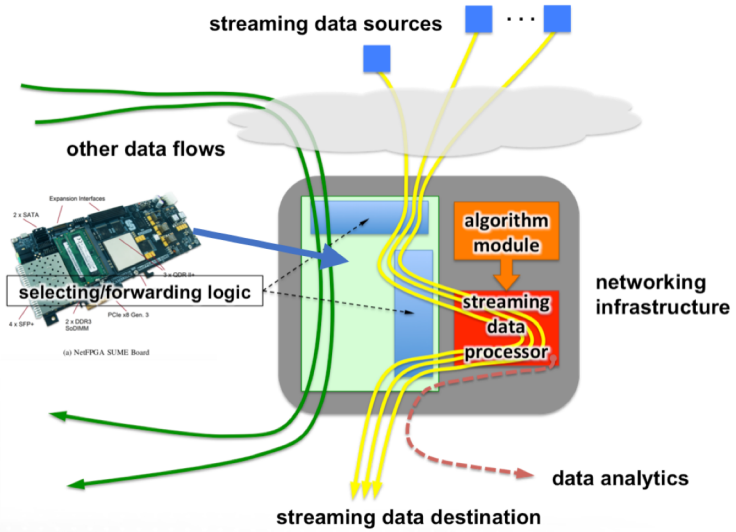
# Our approach

- Move data streaming process to **network functions** or **devices**
- Adopt software defined network (SDN) - manage heterogeneous devices and upgrade control strategies
  - Separate the control and data planes while using centralized, software-based hierarchical control logic
- Use programmable hardware (NetFPGA) to upgrade network

# Network architecture

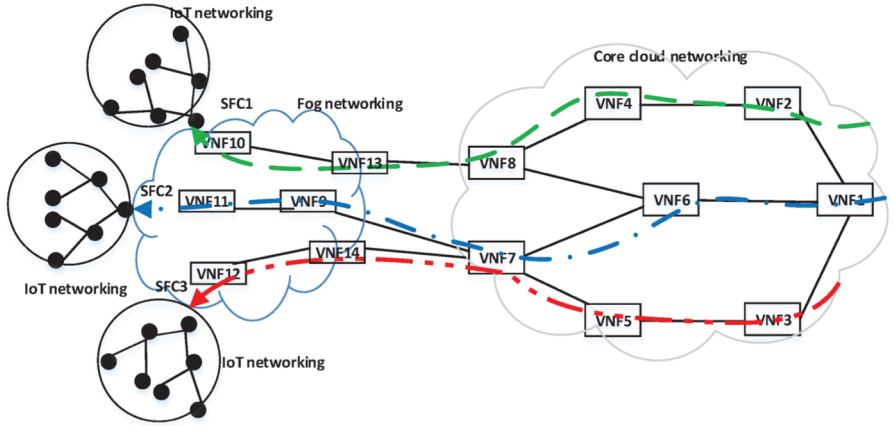
- Logical control is a tree-based hierarchical architecture
- Core controller maintains global network information
  - topology, historical data, data locations, global network parameters and policies, etc.
  - Initiate infrequent exchange of control msg
- Controllers and programmable switches cooperate with each other
- Besides forwarding, programmable switches can run some light weight streaming data processing
  - Generic to streaming algorithms
  - Aggregation, labeling, sampling, simple statistics, cleaning data, monitoring



# In-network computation

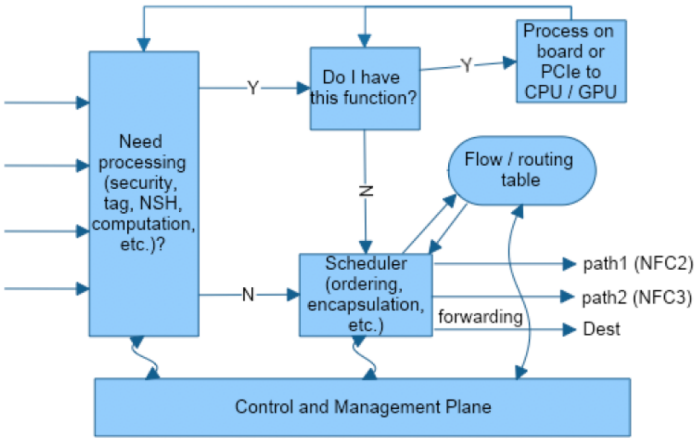
- **M**ethod 1: Use network function chain (NFC) from SDN
  - Design SDN switch, especially for data plane
  - Use network service header (NSH) to realize tunneling




- **M**ethod 2: Process data on the switch (future work, achievable)
  - i.e. Barefoot's Tofino and Deep Insight (monitor network flows)

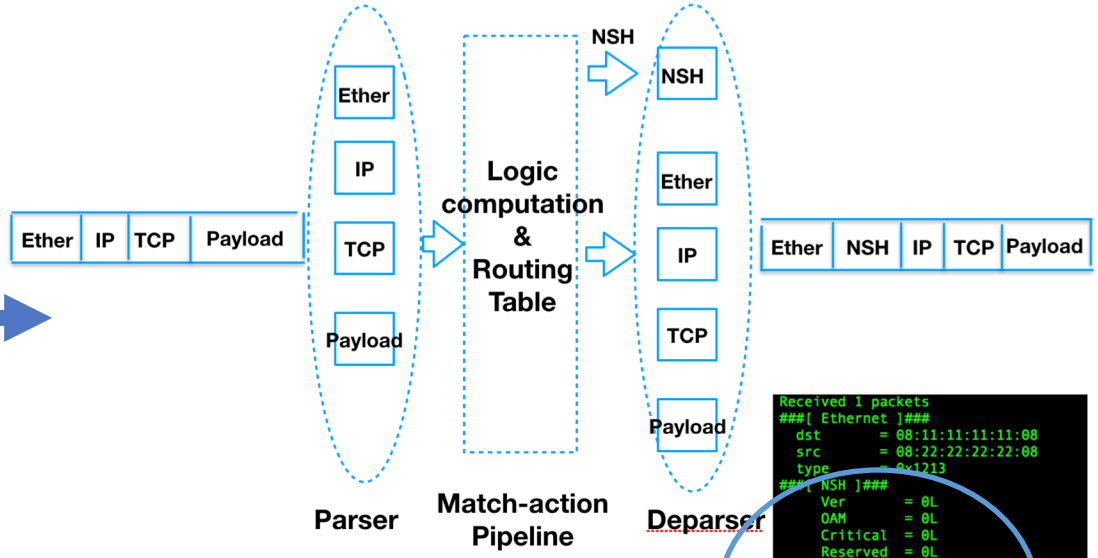
# Implement NSH in NetFPGA switch

## Logical control flow




Switch

## SimpleSumeSwitch Architecture

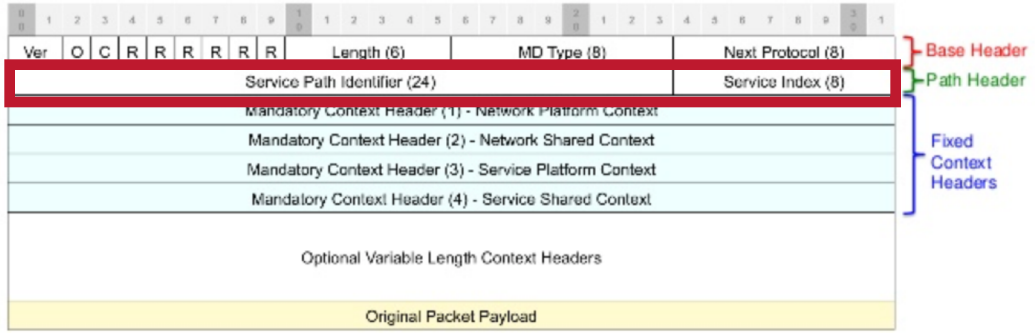


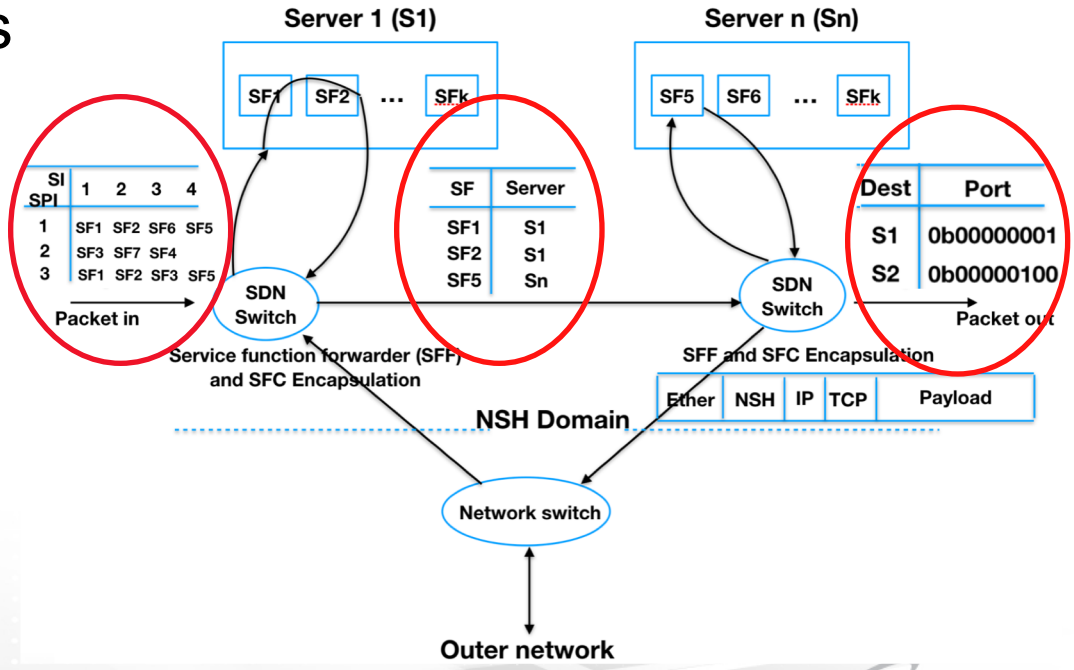

```
tcpdump: listening on eth1, link-type EN10MB (
The HW testing tool for the switch_calc design
 type help to see all commands
[testing> run_test 9 - 6
Finished to send 1 packets
###[ Ethernet ]###
  dst       = 08:22:22:22:22:08
  src       = 08:11:11:11:11:08
  type      = 0x1212
###[ Calc ]###
     op1       = 9
     opCode    = SUB
     op2       = 6
     result    = 0
###[ Raw ]###
        load      = '\x00\x00\x00\x00\x00\x00\
```

```
Received 1 packets
###[ Ethernet ]###
  dst       = 08:11:11:11:11:08
  src       = 08:22:22:22:22:08
  type      = 0x1213
###[ NSH ]###
     Ver       = 0L
     OAM       = 0L
     Critical  = 0L
     Reserved  = 0L
     Len       = 0L
     MDType    = Fixed Length
     NextProto = Ethernet
     NSP       = 5
     NSI       = 5
     NPC       = 0
     NSC       = 0
     SPC       = 0
     SSC       = 0
###[ Calc ]###
        op1       = 9
        opCode    = SUB
        op2       = 6
        result    = 3
###[ Raw ]###
           load      = '\x00\x00\x00\
```

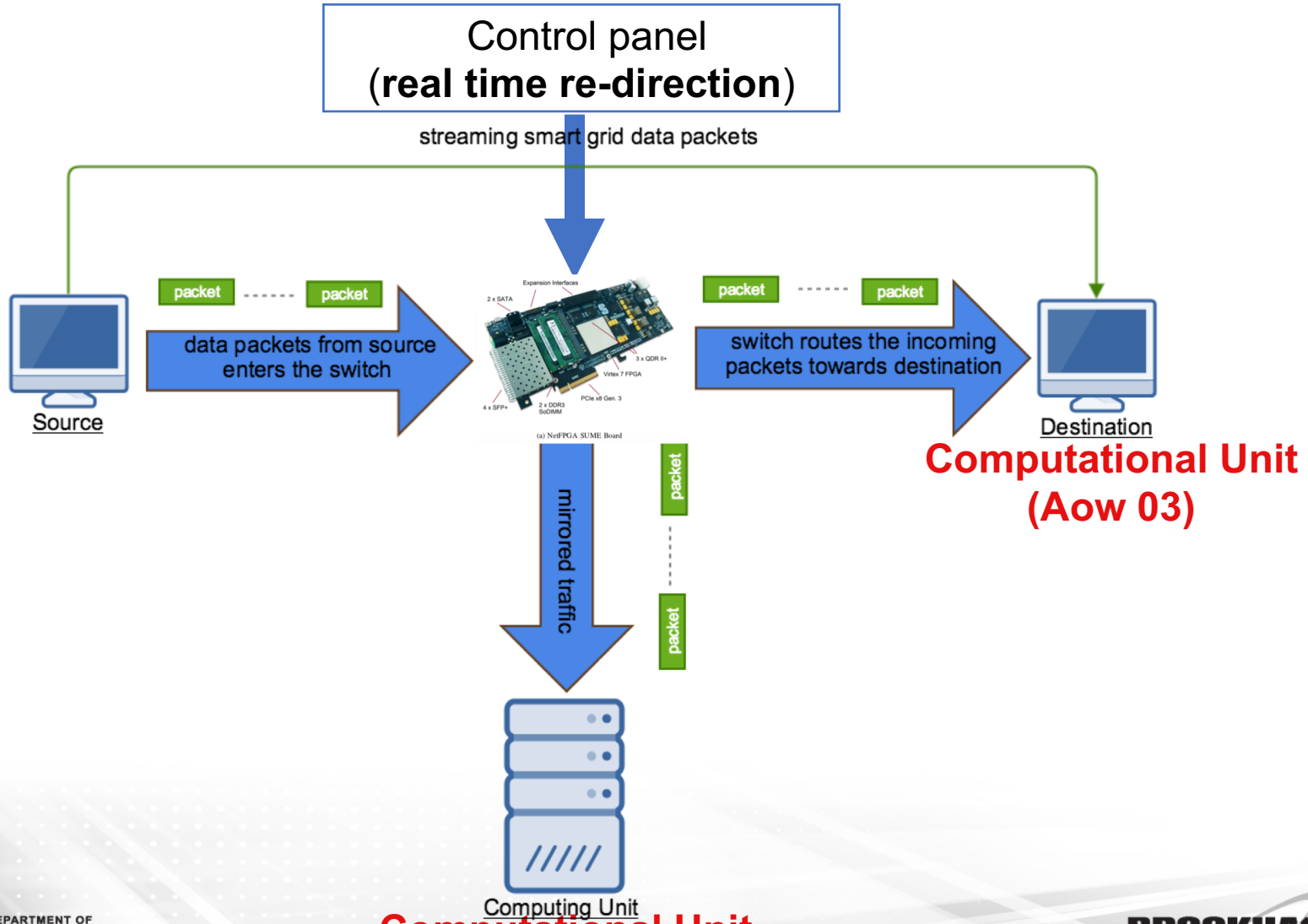
# NSH routing by NetFPGA switch

- Service Path Identifier (SPI): uniquely identifies a service function path (SFP)
  - SPI 1: |SF1| <—> |SF2| <—> |SF4| <—> |SF5|
- Service Index (SI): provides location within the SFP
- In NetFPGA we implement **three match-action tables** to route packets through network service function
  - SPI-SI table -> SF table -> forwarding table

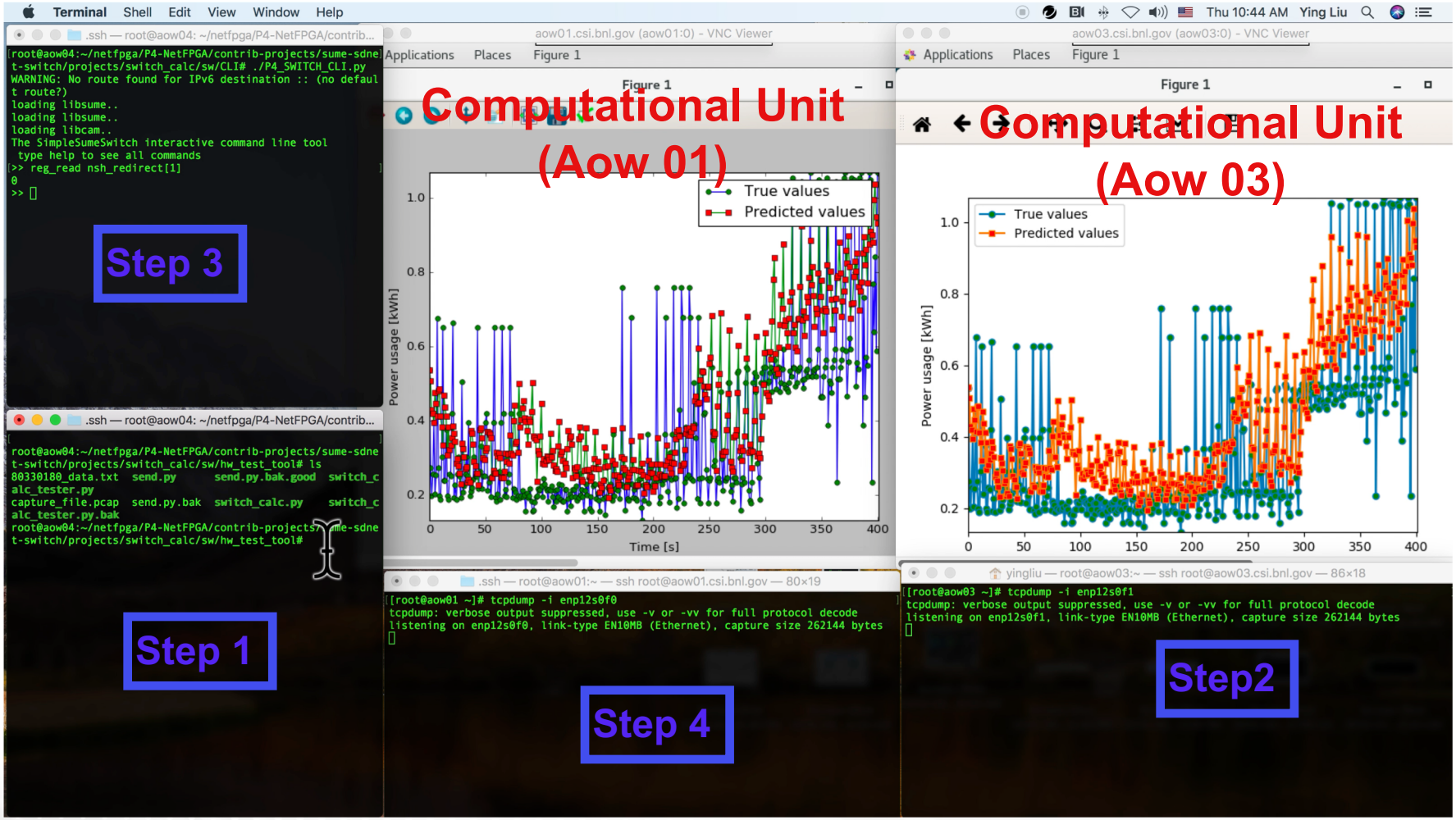


# Testbed procedures - redirect packet flows



# Demo - redirect packet flows



# Challenges: in-network computation

- **Programmable data plane hardware -> opportunity to reconsider division of computation**
- **What kind of computation should be delegated to network?**
  - Monitoring data flows
  - Data aggregation, correlation, compression/decompression, etc.
  - Calculate flow statistics, i.e. min, max, expectation, flow size, probability distribution
  - Interpolation or/and extrapolation of data
  - Clean data for machine learning
  - Run simple machine learning algorithms
- **Goals/Requirements**
  - Reduce: application runtime, load on servers, network congestion
  - Increase: application scalability

# Challenges: executing data analysis on switch

- Difficult to deeply explore **spatial** and **long time auto-correlation** unless the device can access buffers at high speed with sufficient storages
- Third-party libraries, written in low-level languages to support **packet content processing** in hardware, are lacking
- In-switch packet **reassembly** must consider **consistency** when reconstructing data from various sources and latency - a basic requirement to analyze data with semantic meaning
- **Data types and operations** currently supported in hardware language (i.e. p4) are simple - **not supporting** taking derivatives and gradients

# Conclusion

- We have discussed an SDN network architecture to support in-network data processing
- Have implemented NSH through designing NetFPGA as a switch to redirect in-network computation

# Future works

- Will support IP and TCP protocol to test consistency of our SDN switch with a conventional switch
- Implement cooperative network controls among SDN switches - distributed
- Implement network resource sharing (with parity)
- Guarantee high-bandwidth data security
- In general, consider the scalability of network architecture for streaming data processing (including reliability)

# Thank you!

# Questions?

**Ying Liu**
**Email: yliu@bnl.gov**