Managing a Virtual Network Function using SDN and Control Theory

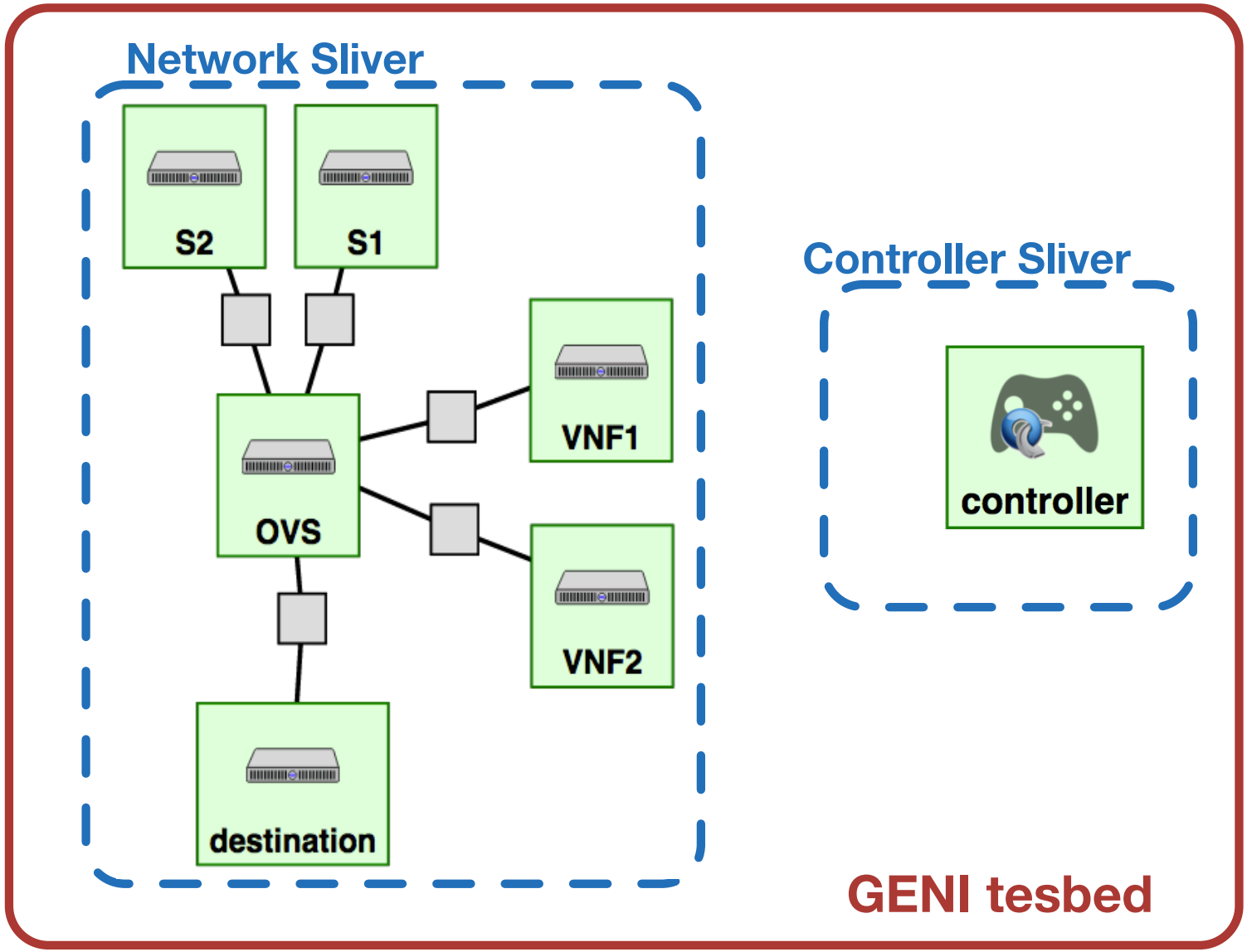
GENI Summer Camp @ U. Kentucky
May 16th, 2018

Ibrahim Matta

Joint work with Nabeel Akhtar

BOSTON UNIVERSITY

GENI resources that we need …



BOSTON UNIVERSITY

So before we get started …

- Login to GENI at http://portal.geni.net and select the GRW-Summer-Camp-UKentucky project
- Create two new slices for network and controller
  - Ryu version: http://tinyurl.com/geninfv
  - Follow Step 3 (Obtain resources: 3.1 & 3.2) under **Design/Setup**

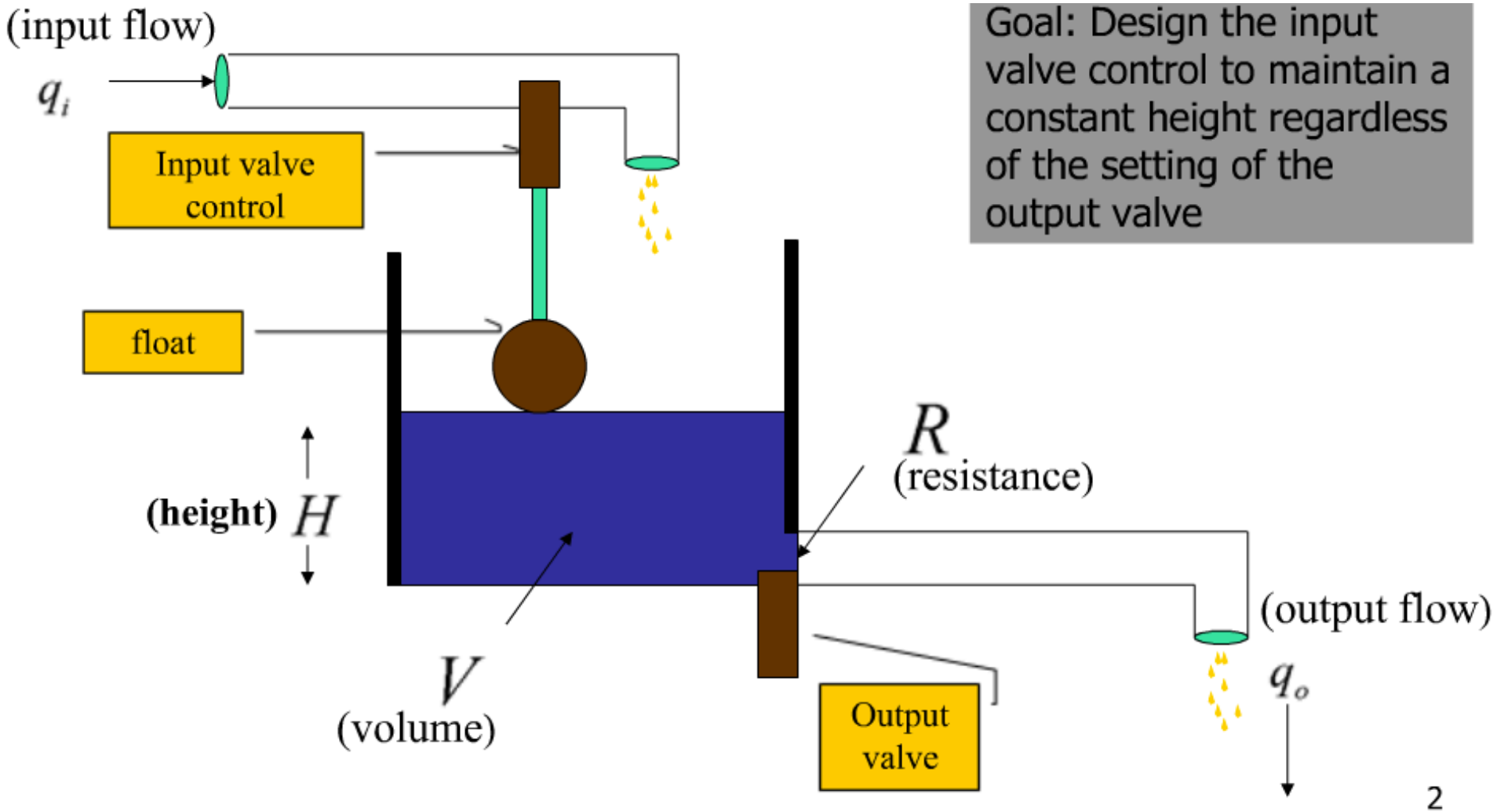
- Bind your resources to an **InstaGENI** rack
- Reserve your resources
- Later we will login to these VMs

BOSTON UNIVERSITY

Control Theory

BOSTON UNIVERSITY

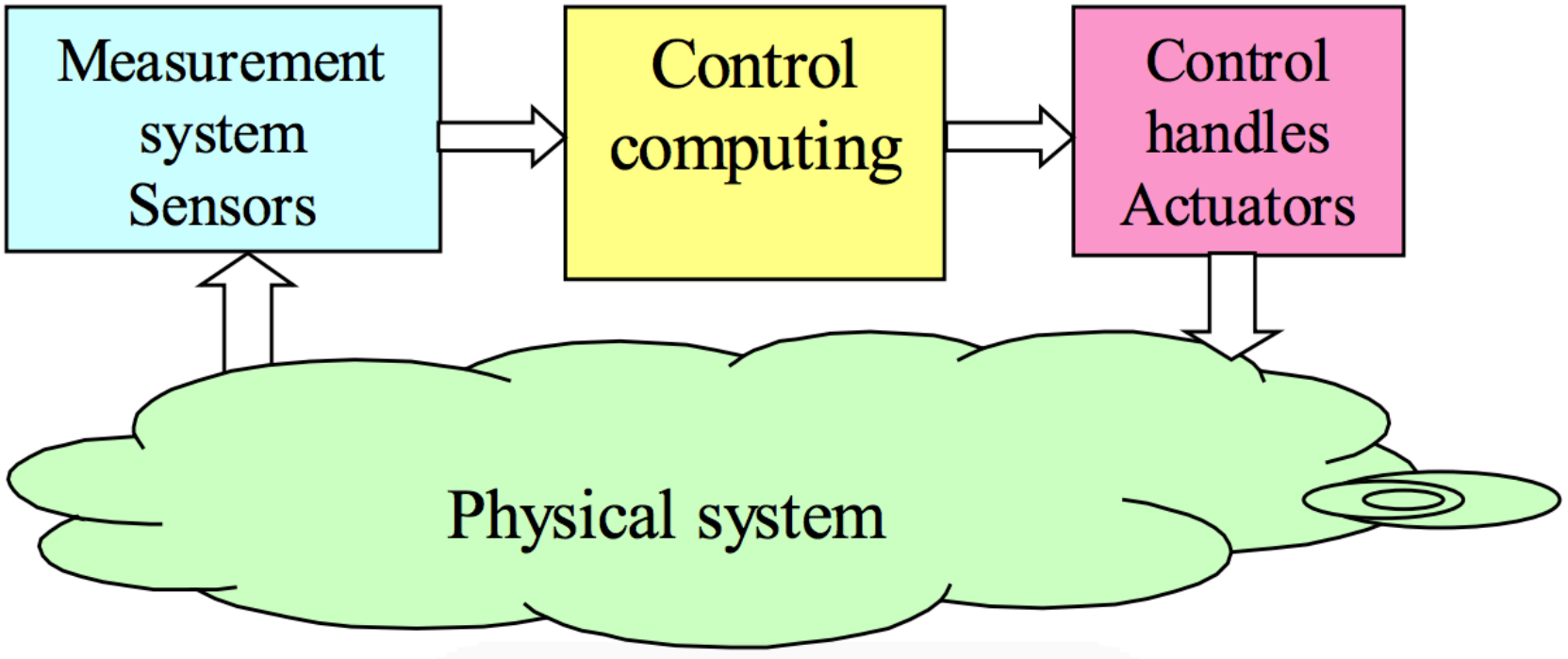
Control Theory



Goal: Design the input valve control to maintain a constant height regardless of the setting of the output valve

BOSTON UNIVERSITY

Control Theory



“90% of the real world applications are based on 10% of the existing control methods and theory”

Dimitry Gorinevsky – Stanford University

BOSTON UNIVERSITY

Examples of Control Theory in CS

- TCP/IP

```
for every loss {
        W = W/2
        }
for every ACK {
        W += 1/W
        }
```

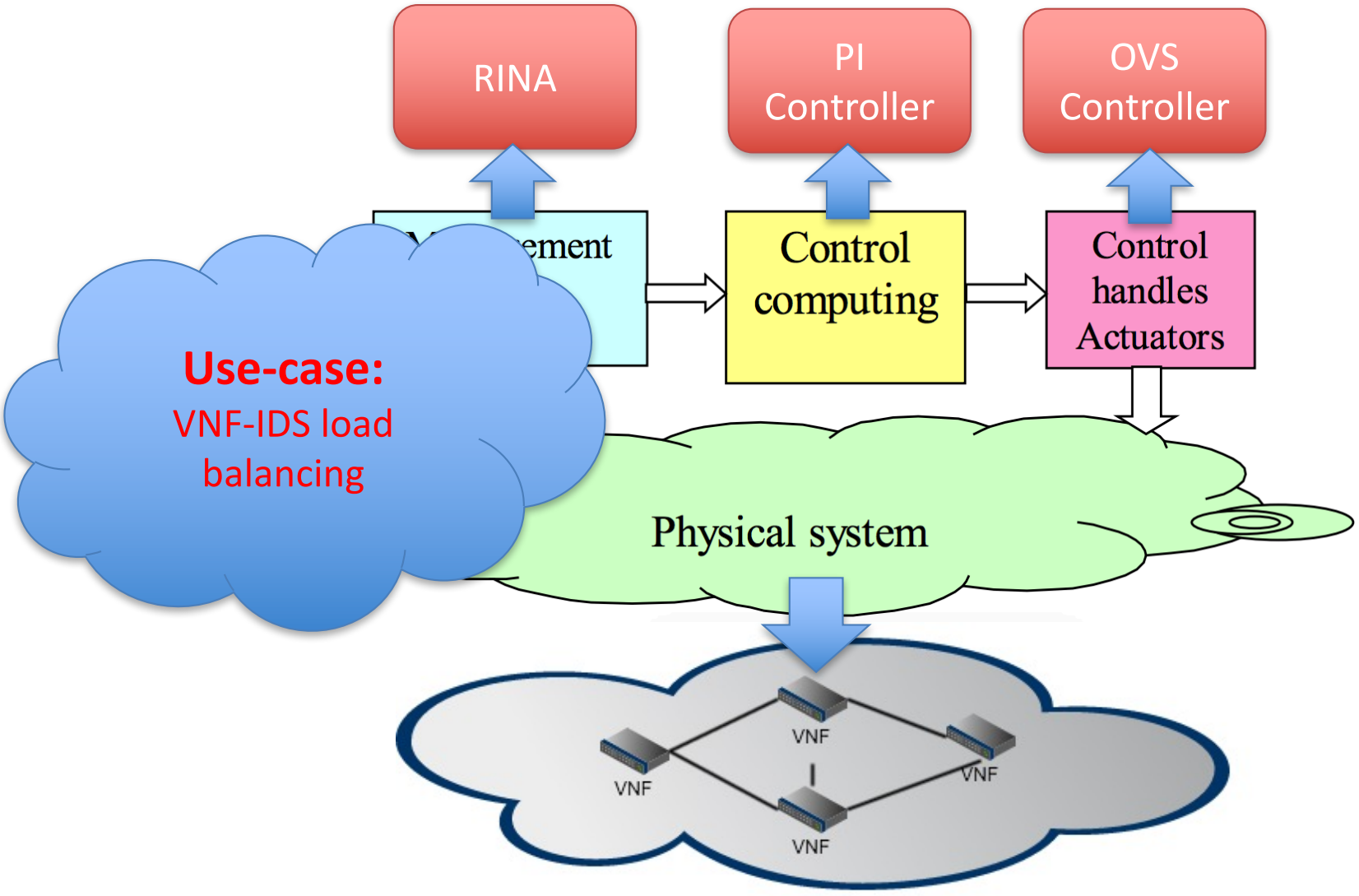
$$\dot{x} = \frac{1-q}{\tau^2} - \frac{1}{2} q x^2$$

- x - transmission rate
- τ - round trip time
- q - loss probability

  - Analysis and systematic design was developed some 20 years later
- QoS in Caching
- Apache QoS differentiation
- …
- See: **Optimizing and Modeling Dynamics in Networks**. In Hamed Haddadi and Olivier Bonaventure, editors, eBook on Recent Advances in Networking, volume 1. ACM SIGCOMM, August 2013.

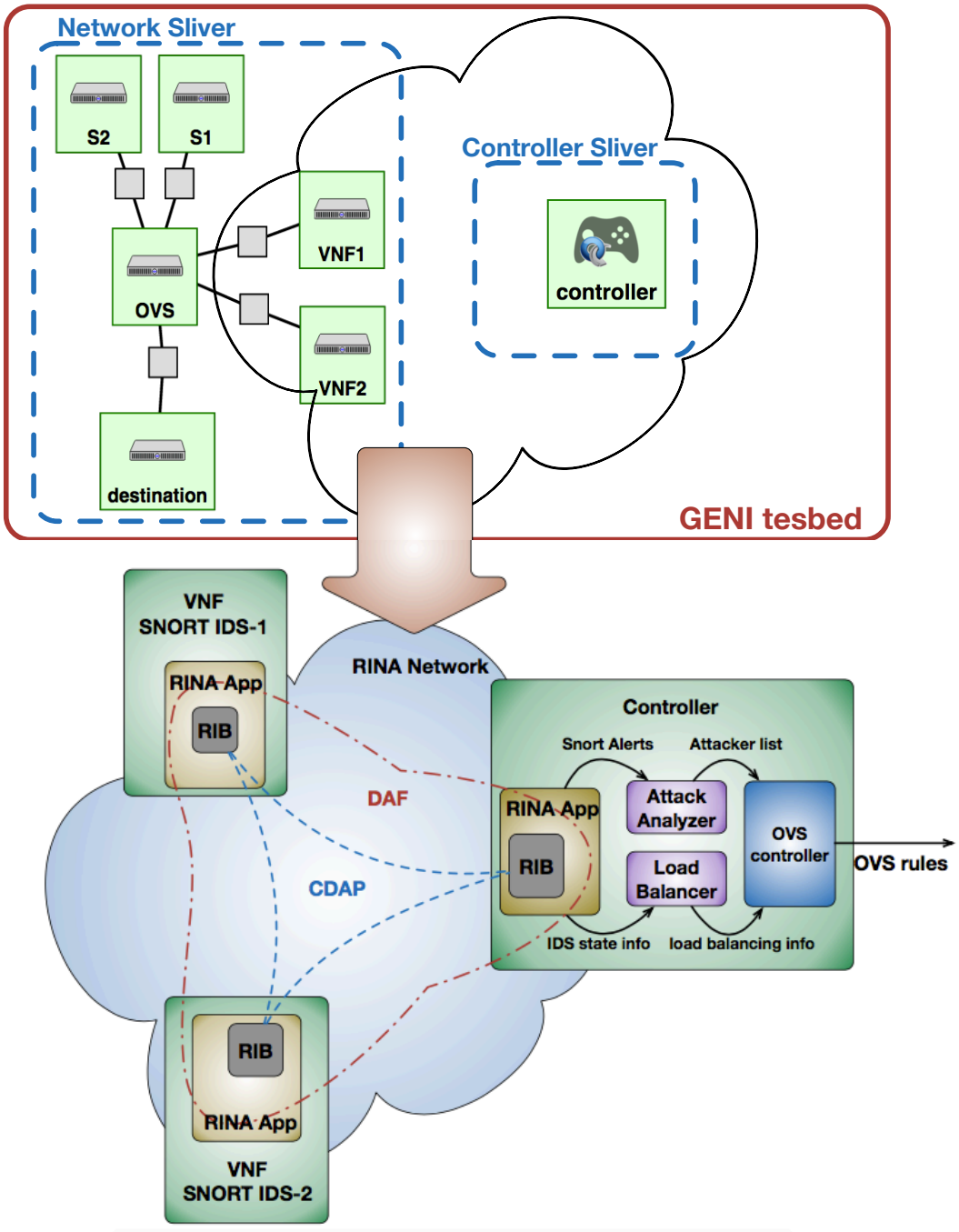
BOSTON UNIVERSITY

Managing NFV using SDN & Control Theory



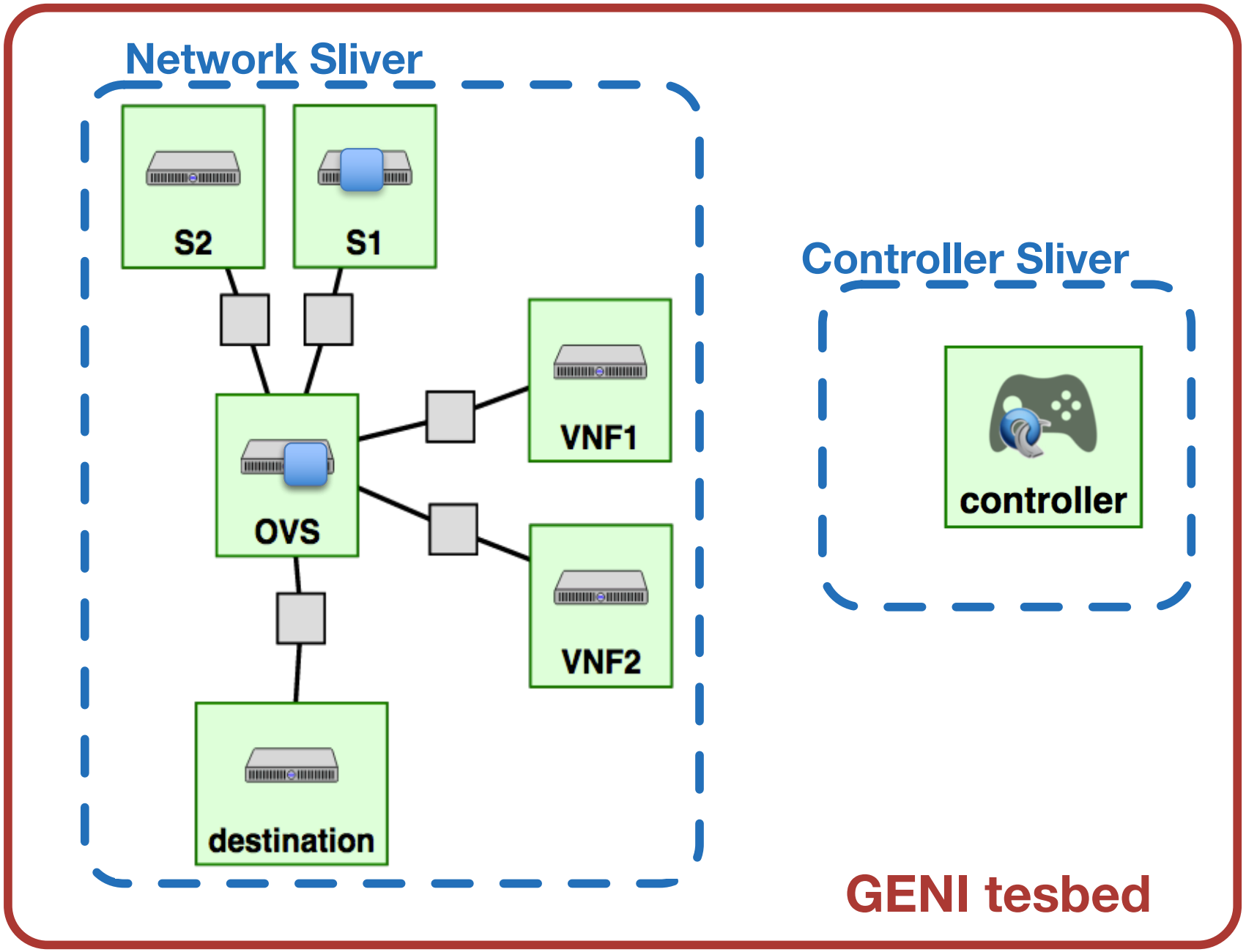
BOSTON UNIVERSITY

System Overview



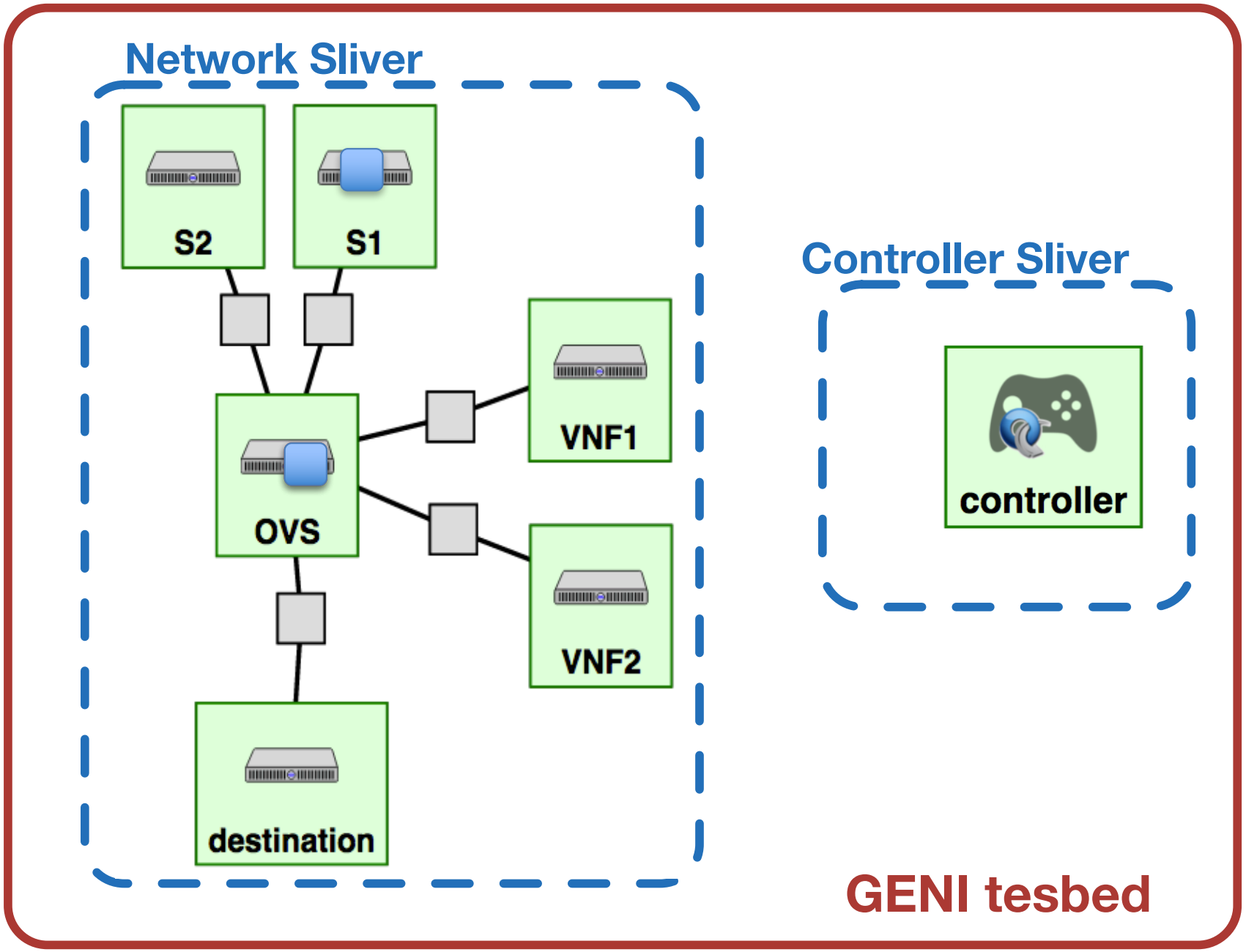
BOSTON UNIVERSITY

Network Traffic



BOSTON UNIVERSITY

Network Traffic



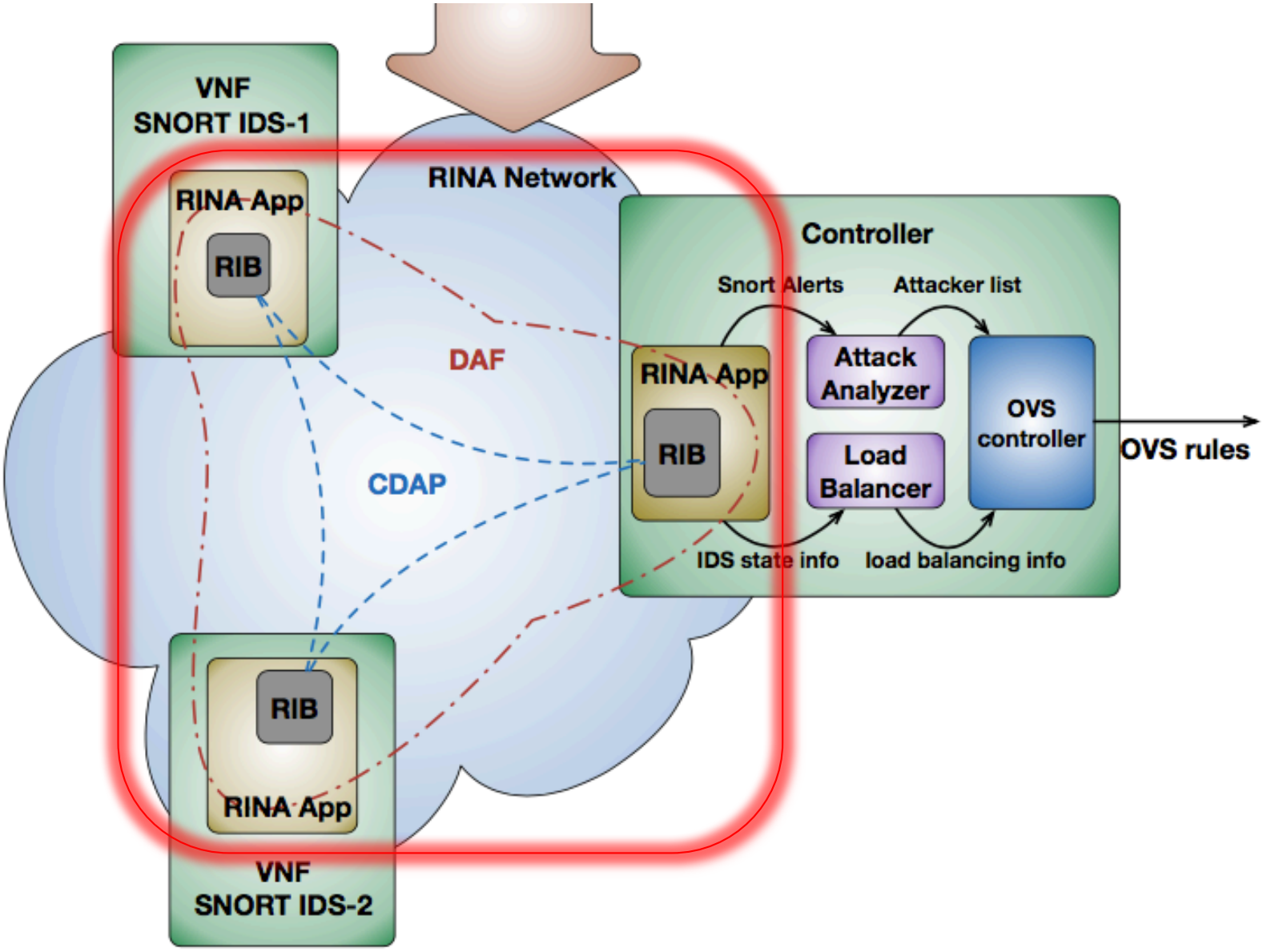
BOSTON UNIVERSITY

Snort as IDS

- Open source IDS system widely deployed
- InfoWorld's Open Source Hall of Fame as one of the "greatest open source software of all time”
- Protocol analysis, content searching and content matching

BOSTON UNIVERSITY

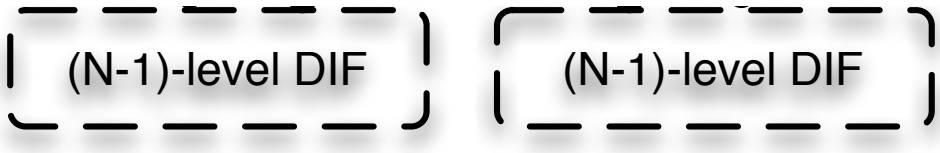
RINA



BOSTON UNIVERSITY

Recursive InterNetwork Architecture (RINA)

- Clean slate Future Internet Architecture
- Networking is Inter-process communication (IPC)
  - Old principle applied (e.g., TCP RFC 793, 1981)
- DIF (Distributed IPC Facility) – processes cooperating to provide IPC
- DAF – processes cooperating to perform a certain function

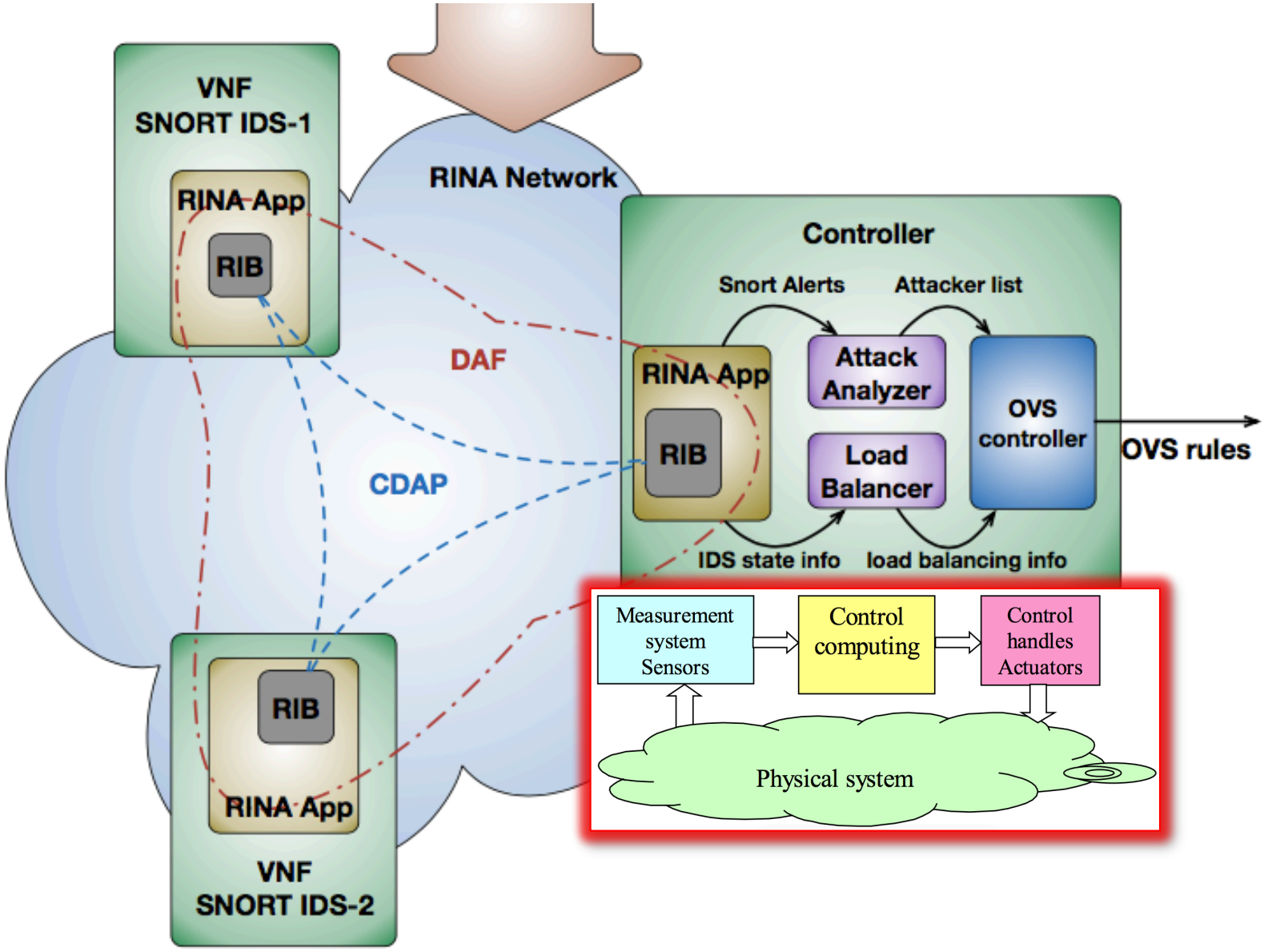


http://csr.bu.edu/rina/

See GEC19 Tutorial: ***www.youtube.com****/watch?v=qUDvduy-JEs*

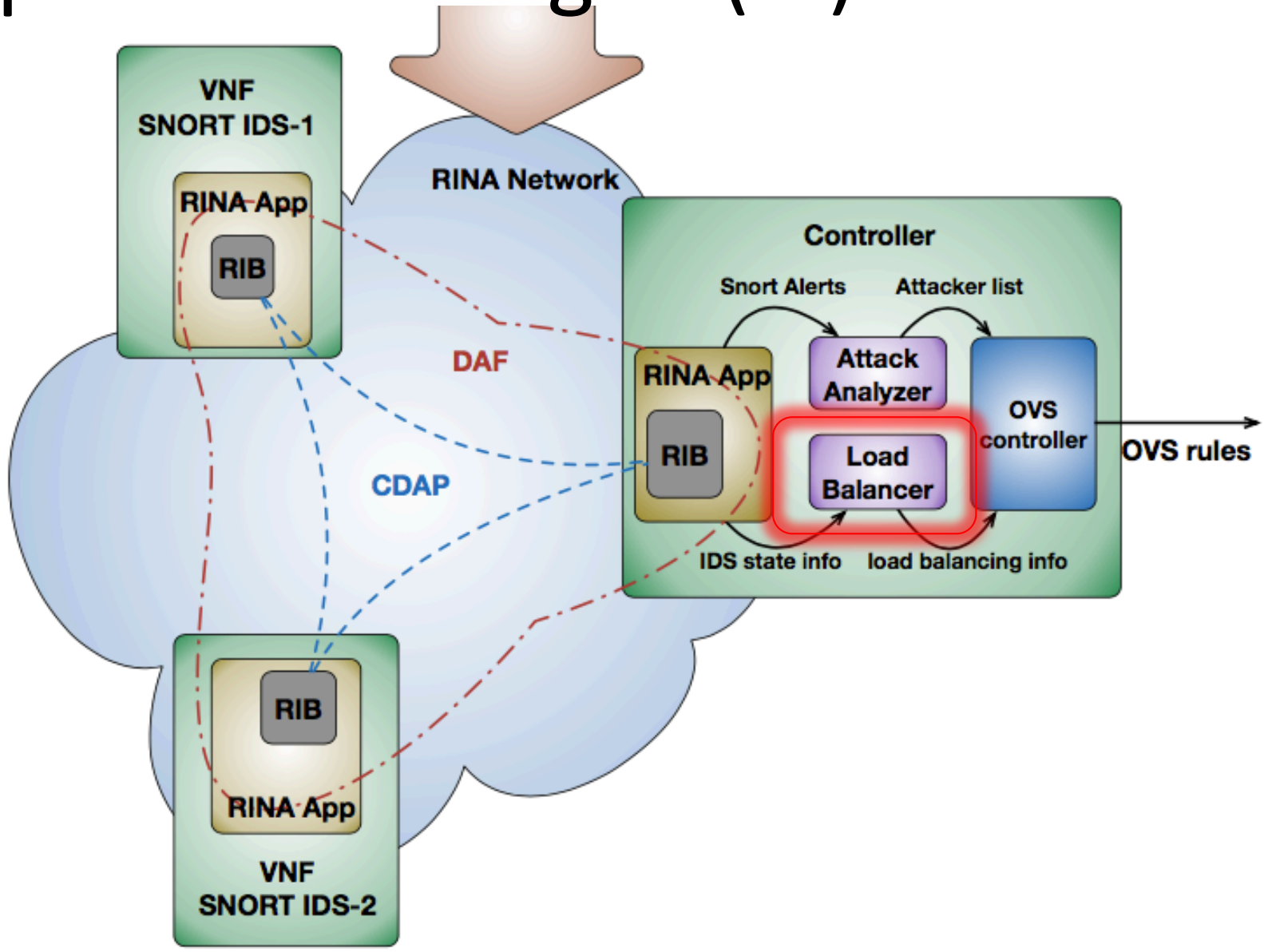
BOSTON UNIVERSITY

Controller



BOSTON UNIVERSITY

Proportional Integral (PI) Controller



BOSTON UNIVERSITY

Proportional Integral (PI) Controller

$$x(t) = \max[0, \min[1, x(t-1) + K(\frac{L(t)}{T} - 1)]]$$

$x(t)$: ratio of traffic diverted to VNF2 at time t

$L(t)$: load on VNF1

T: target load on VNF1

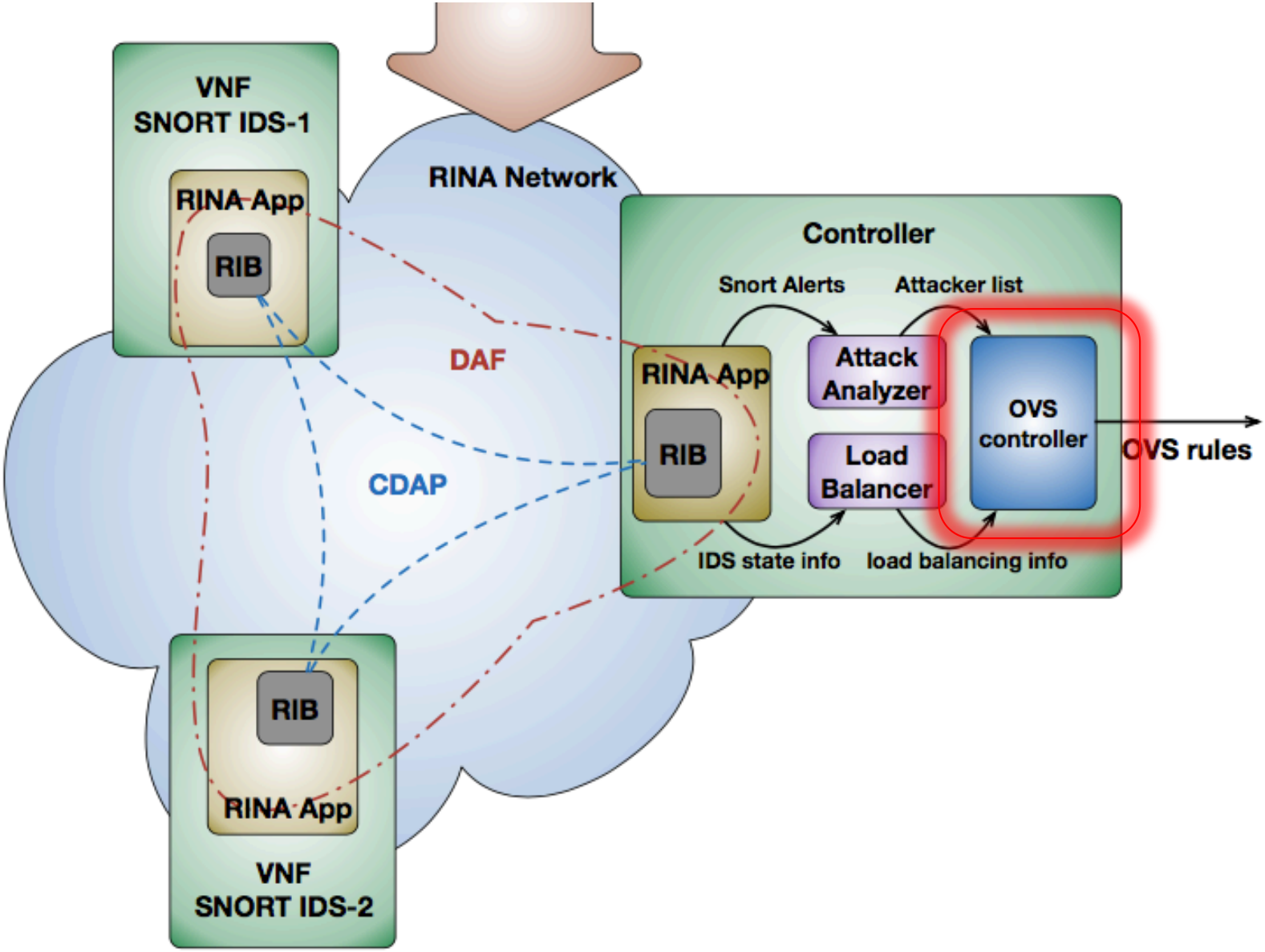
Algorithm 1 PI controller

Input: $IDS_{load}.txt$
Output: $x(t)$

1: $T = 0.5$
2: $x(t-1) = 0.0$
3: $x(t) = 0.0$
4: $K = 0.2$
5: **while** $True$ **do**
6: $\quad L(t) = getLoad(IDS_{load}.txt)$;
7: $\quad x(t) = \max[0, \min[1, x(t-1) + K(\frac{L(t)}{T} - 1]]$;
8: $\quad write(t, x(t))$;
9: **end while**

BOSTON UNIVERSITY

PI-based OVS Controller



BOSTON UNIVERSITY

PI-based OVS Controller

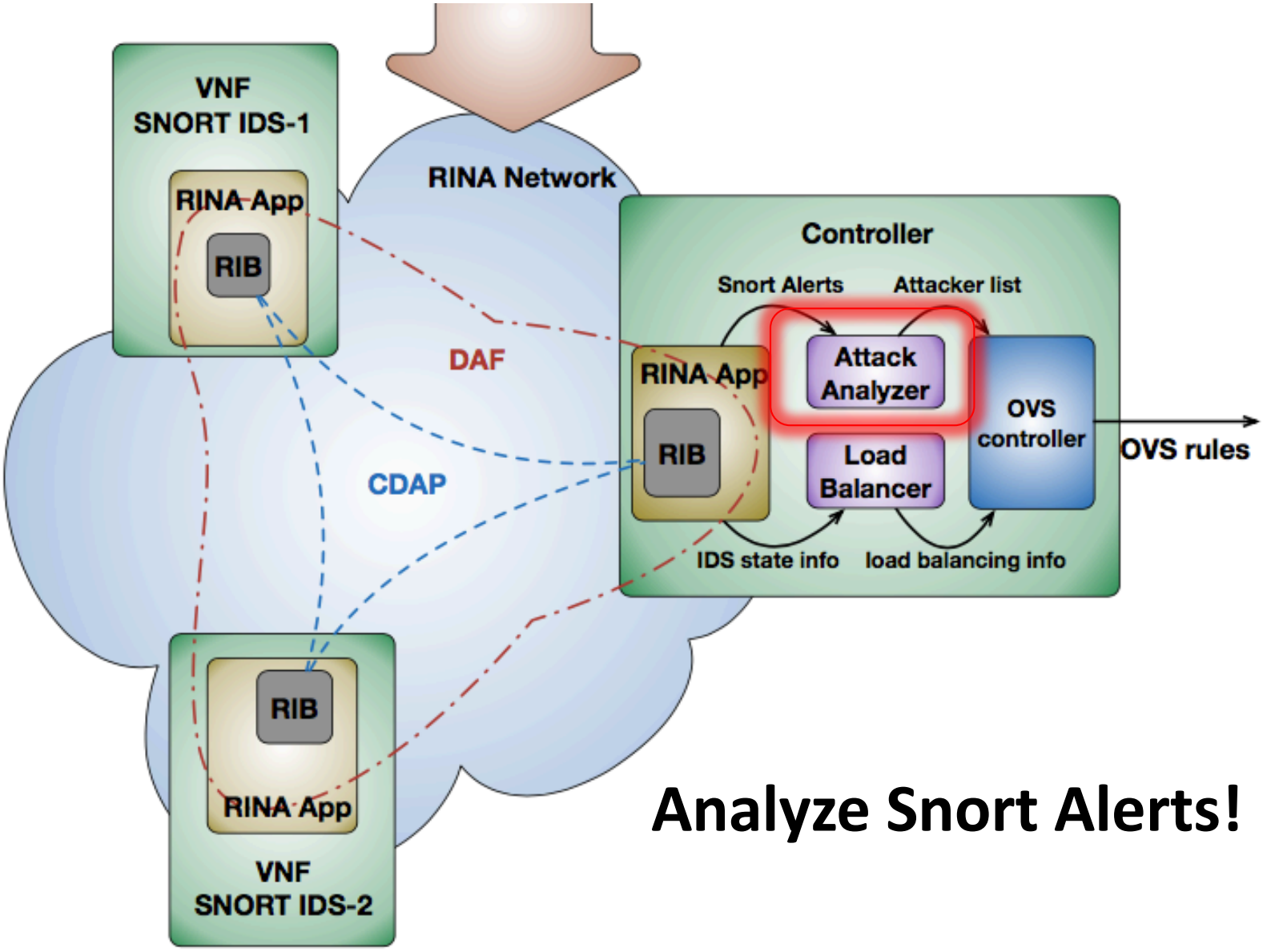
Algorithm 2 PI-based OVS controller

Input: $Flows,\ x(t)$

1: **for all** f *in* $Flows$ **do**
2: $random = generateRandom();$
3: **if** $random > x(t)$ **then**
4: $vnfSelected = IDS1;$
5: **else**
6: $vnfSelected = IDS2;$
7: **end if**
8: $sendFlow(f,\ vnfSelected);$
9: **end for**

BOSTON UNIVERSITY

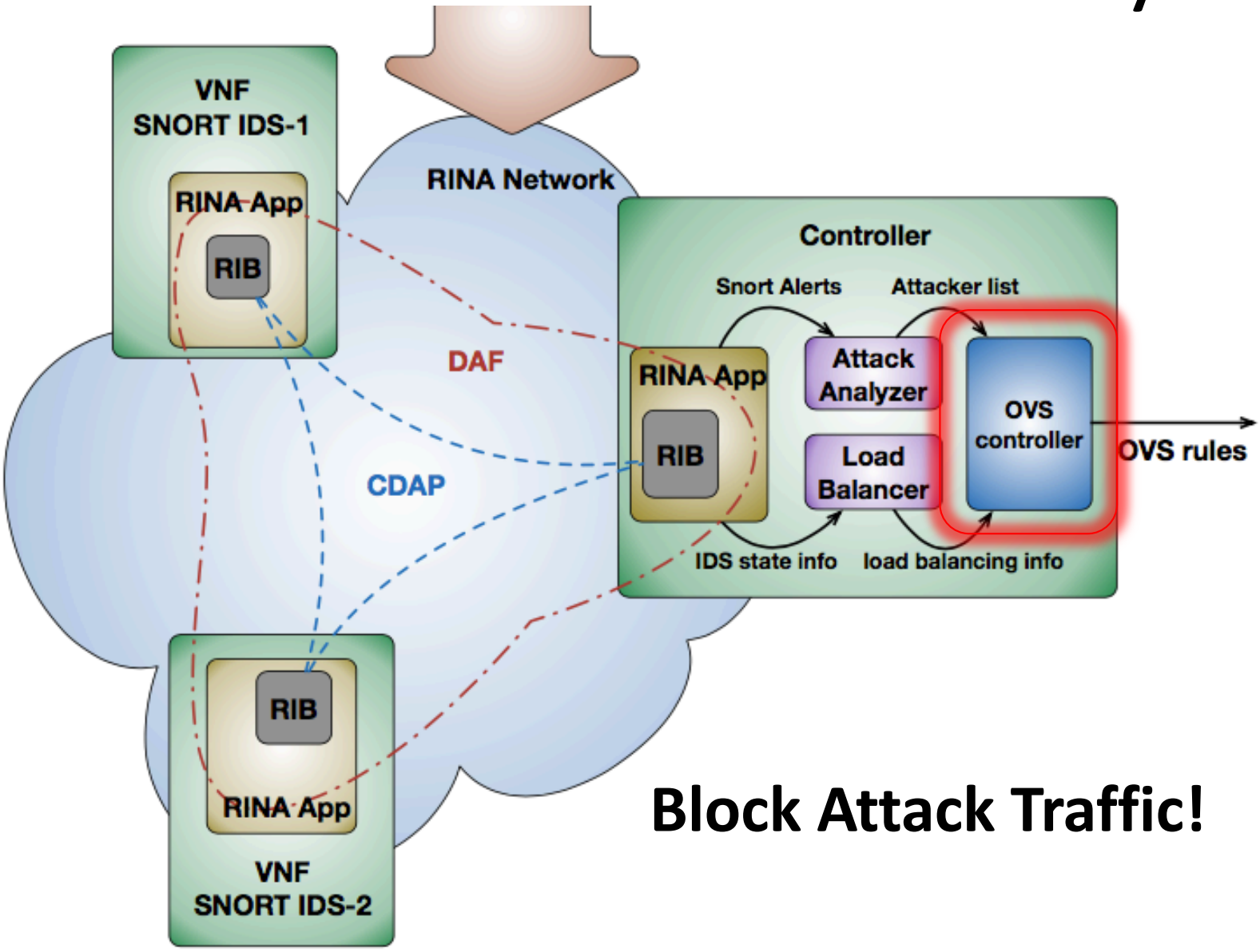
Attack Analyzer



Analyze Snort Alerts!

BOSTON UNIVERSITY

OVS controller with Attack Analyzer



Block Attack Traffic!

BOSTON UNIVERSITY

Load Balancer: Round Robin based OVS Controller

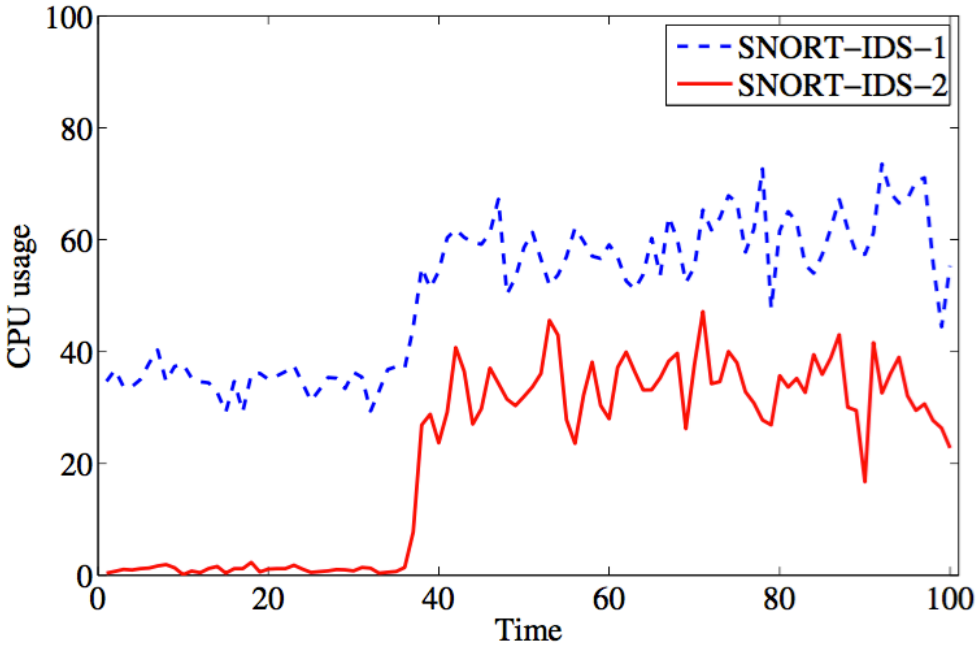
Algorithm 3 Round Robin based OVS controller

Input: $Flows$

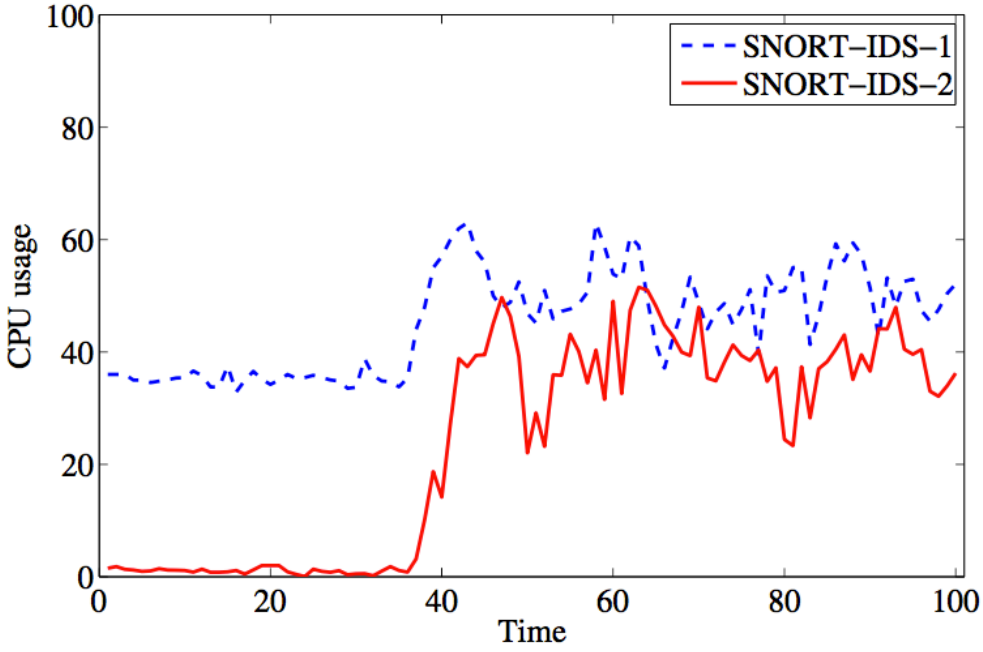
```
1: vnfSelected = IDS1
2: for all f in Flows do
3:     if vnfSelected == IDS1 then
4:         vnfSelected = IDS2;
5:     else
6:         vnfSelected = IDS1;
7:     end if
8:     sendFlow(f, vnfSelected);
9: end for
```

BOSTON UNIVERSITY

Load Balancer: Round Robin vs PI Control



Simple Round Robin load balancing



Load balancing based on PI control (T = 50%)

BOSTON UNIVERSITY

Load Balancer:

Load balancer vs No load balancer

- Attacks are detected significantly faster with load balancer

```
nabeel@s1:~$ cat ping.log
PING destination-lan4 (10.10.1.5) 56(84) bytes of data.
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=1 ttl=64 time=8.23 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=2 ttl=64 time=70.0 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=3 ttl=64 time=211 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=4 ttl=64 time=8.65 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=5 ttl=64 time=9.03 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=6 ttl=64 time=8.95 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=7 ttl=64 time=10.4 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=8 ttl=64 time=321 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=9 ttl=64 time=338 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=10 ttl=64 time=394 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=11 ttl=64 time=319 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=12 ttl=64 time=126 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=13 ttl=64 time=39.8 ms

--- destination-lan4 ping statistics ---
422 packets transmitted, 13 received, 96% packet loss, time 87494ms
rtt min/avg/max/mdev = 8.231/143.584/394.124/145.405 ms, pipe 2
```

a)

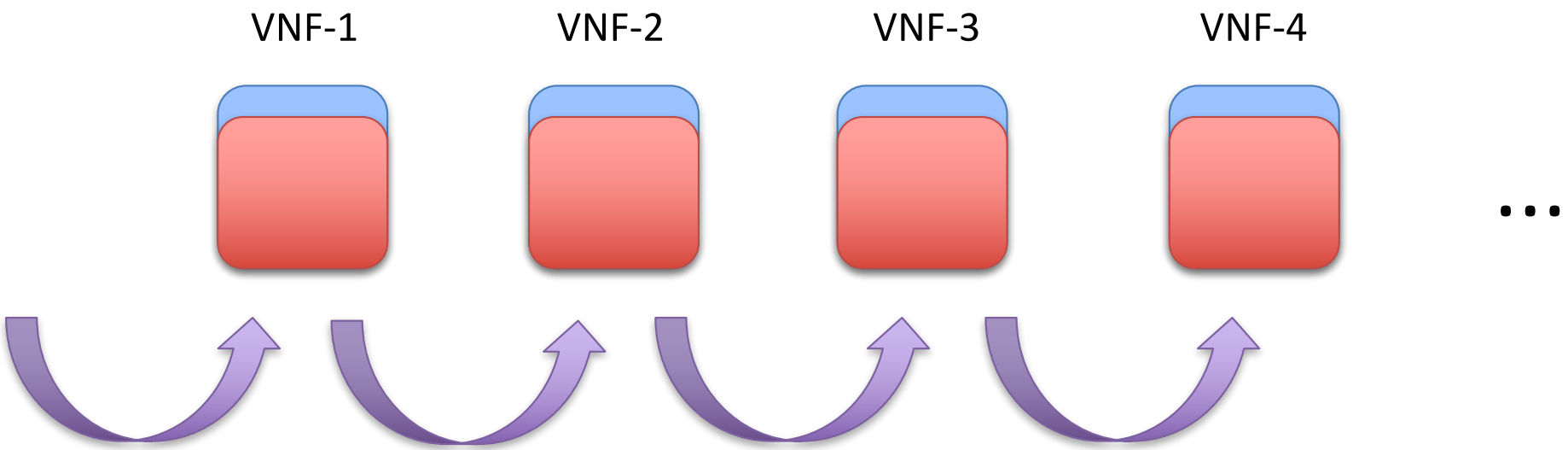
```
nabeel@s2:~$ cat ping.log
PING destination-lan4 (10.10.1.5) 56(84) bytes of data.
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=1 ttl=64 time=10.0 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=2 ttl=64 time=80.1 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=3 ttl=64 time=186 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=4 ttl=64 time=53.4 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=5 ttl=64 time=9.67 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=6 ttl=64 time=10.0 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=7 ttl=64 time=9.69 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=8 ttl=64 time=9.76 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=9 ttl=64 time=359 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=10 ttl=64 time=397 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=11 ttl=64 time=273 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=12 ttl=64 time=73.0 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=13 ttl=64 time=28.1 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=14 ttl=64 time=15.4 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=15 ttl=64 time=18.1 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=16 ttl=64 time=16.5 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=17 ttl=64 time=46.7 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=18 ttl=64 time=17.7 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=19 ttl=64 time=11.4 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=20 ttl=64 time=17.1 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=21 ttl=64 time=17.8 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=22 ttl=64 time=13.1 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=23 ttl=64 time=36.7 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=24 ttl=64 time=55.8 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=25 ttl=64 time=8.30 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=26 ttl=64 time=13.7 ms
64 bytes from destination-lan4 (10.10.1.5): icmp_seq=27 ttl=64 time=14.6 ms

--- destination-lan4 ping statistics ---
161 packets transmitted, 27 received, 83% packet loss, time 33098ms
rtt min/avg/max/mdev = 8.302/66.844/397.653/105.658 ms, pipe 2
```

b)

Port Scanning Attack (a) 2.6 seconds with load-balancer (b) 5.4 seconds without load-balancer

BOSTON UNIVERSITY

Scaling



BOSTON UNIVERSITY

DEMO

Experiment 3: Handling Intrusion with Ryu Controller: Ping Attack

BOSTON UNIVERSITY

To Do:

Go to tutorial http://tinyurl.com/geninfv and do following steps:

1. Design/Setup
2. Execute
3. Experiments:
   - **Experiment 1**: Load Balancing using Round Robin Control with Ryu Controller
   - **Experiment 2**: Load Balancing using Proportional Integral (PI) Control with Ryu Controller
   - **Experiment 3**: Handling Intrusion with Ryu Controller: Ping Attack
   - **Experiment 4**: Handling Intrusion with Ryu Controller: Port Scanning Attack

BOSTON UNIVERSITY

To Do:

Go to tutorial http://tinyurl.com/geninfv and do following steps:

1. Design/Setup
2. Execute
3. Experiments:
   - **Experiment 1**: Load Balancing using Round Robin Control with Ryu Controller
   - **Experiment 2**: Load Balancing using Proportional Integral (PI) Control with Ryu Controller
   - **Experiment 3**: Handling Intrusion with Ryu Controller: Ping Attack
   - **Experiment 4**: Handling Intrusion with Ryu Controller: Port Scanning Attack

BOSTON UNIVERSITY

Conclusion

- First work that combines Control Theory with SDN/NFV management
- Control Theory can play crucial role in SDN/NFV management
- Use case: Load balancer for IDS (VNF)
  - GENI test-bed is used for realistic experimentation
  - RINA based distributed application is used for monitoring
  - PI-Controller
  - Scaling
- Attacks detected faster with load balancer!

BOSTON UNIVERSITY

Tutorial to reproduce results:

POX version: http://groups.geni.net/geni/wiki/GENIExperimenter/Tutorials/NFV

Ryu version: http://groups.geni.net/geni/wiki/GENIExperimenter/Tutorials/NFV/Ryu

Our papers:

Nabeel Akhtar, Ibrahim Matta and Yuefeng Wang. "Managing NFV using SDN and Control Theory". IEEE/IFIP International Workshop on Management of the Future Internet (ManFI 2016), co-located with NOMS 2016, Istanbul, Turkey, April 2016.
[Technical Report: http://csr.bu.edu/rina/papers/BUCS-TR-2015-013.pdf]
[Demo at NOMS 2016: http://csr.bu.edu/rina/papers/NOMS2016Demo.pdf]

Nabeel Akhtar, Ibrahim Matta, Ali Raza and Yuefeng Wang. **EL-SEC: ELastic Management of SECurity Applications on Virtualized Infrastructure** got accepted at **IEEE INFOCOM** International Workshop on Computer and Networking Experimental Research Using Testbeds (**CNERT**) 2018, Hawaii, USA. https://github.com/akhtarnabeel/ELSEC
[Demos at CNERT and IEEE INFOCOM 2018]