

# Software Defined Networking

## Lab Work 3 Introduction



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

Jeremias Blendin, Leonhard Nobach, Christian Koch,  
Julius Rückert, Matthias Wichtlhuber



PS - Peer-to-Peer Systems Engineering Lab  
Dept. of Electrical Engineering and Information Technology  
Technische Universität Darmstadt  
Rundeturmstr. 12, D-64283 Darmstadt, Germany  
<http://www.ps.tu-darmstadt.de/>

# Lab Work 3

---



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

## Exploring OpenFlow 1.3

# Run a simple\_switch with OpenFlow 1.0



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

## ❖ Test with OpenFlow 1.0

- First terminal: ssh session 1:

```
~$ ryu-manager ryu.app.simple_switch
```

- Second terminal: ssh session 2

```
~$ mn --topo single,3 --mac --arp --switch ovsk \  
    --controller=remote,ip=127.0.0.1  
mininet> h1 ping h2
```

- Third terminal: ssh session 3

- Note the new tool: ovs-ofctl (google „man ovs-ofctl“ for details)

```
~$ sudo ovs-ofctl dump-flows s1
```

## ❖ Note the debug output in session 1

- Investigate the source code
- Find out the meaning of the log messages

# Run a simple\_switch with OpenFlow 1.3



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

## ❖ Test with OpenFlow 1.3

- First terminal: ssh session 1:

```
~$ ryu-manager ryu.app.simple_switch_13
```

- Second terminal: ssh session 2

```
~$ mn --topo single,3 --mac --arp \  
    --switch ovsk,protocols=OpenFlow13 \  
    --controller=remote,ip=127.0.0.1  
mininet> h1 ping h2
```

- Third terminal: ssh session 3

```
~$ sudo ovs-ofctl dump-flows s1  
2014-11-24T11:47:01Z|00001|vconn|WARN|unix:/var/run/openvswitch/s1.mgmt: version negotiation  
failed (we support version 0x01, peer supports version 0x04)  
ovs-ofctl: s1: failed to connect to socket (Broken pipe)
```

# Run a simple\_switch with OpenFlow 1.3



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

## ➤ Third terminal: ssh session 3

```
~$ sudo ovs-ofctl dump-flows s1
```

```
2014-11-24T11:47:01Z|00001|vconn|WARN|unix:/var/run/openvswitch/s1.mgmt: version negotiation failed (we support version 0x01, peer supports version 0x04)  
ovs-ofctl: s1: failed to connect to socket (Broken pipe)
```

## ➤ Version negotiation failed

- We (ovs-ofctl) support version 0x01 (OpenFlow 1.0)
- Peer (Open vSwitch) supports version 0x04 (OpenFlow 1.3)
- ➡ note the version notation

```
~$ sudo ovs-ofctl dump-flows -O Openflow13 s1
```

```
OFFST_FLOW reply (OF1.3) (xid=0x2):  
cookie=0x0, duration=13.922s, table=0, n_packets=13, n_bytes=1274, priority=1,in_port=2,dl_dst=00:00:00:00:00:01 actions=output:1  
cookie=0x0, duration=12.924s, table=0, n_packets=12, n_bytes=1176, priority=1,in_port=1,dl_dst=00:00:00:00:00:02 actions=output:2  
cookie=0x0, duration=14.541s, table=0, n_packets=6, n_bytes=588, priority=0 actions=CONTROLLER:65535
```

## ➤ Specify the OpenFlow version "-O OpenFlow1X"

# Task 1



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

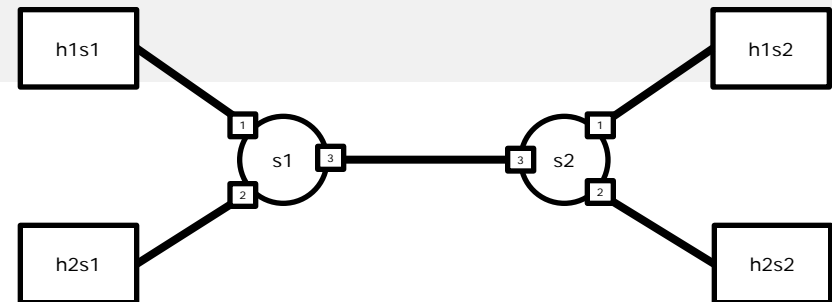
- ❖ Look at the source code of the Ryu modules `simple_switch.py` and `simple_switch_13.py`.
  - Describe and explain the differences between the OpenFlow 1.0 and OpenFlow 1.3 version of `simple_switch`.

# Task 2 (1)

## 1. Have a look at the OpenFlow 1.0-based filtering switch `simple_switch_filter.py`.

1. The basic forwarding method is again layer 2 switching.
2. We increase security by allowing only one host per port
3. This rule does not apply to inter-switch links
4. Run the topology:

```
sudo mn --topo linear,n=2,k=2 --mac --arp --switch ovsk \  
      --controller=remote,ip=127.0.0.1  
mininet> h1s1 ping h2s2
```



## 2. Verify

1. Install and run MAC spoofing tool

```
sudo apt-get update && sudo apt-get install nmap  
mininet> h1s1 nmap --spoof-mac 00:00:00:00:00:33 10.0.0.2
```

2. You should see the following message in your controller output:

```
dropping spoofed packet on s1 src=00:00:00:00:00:33 dst=ff:ff:ff:ff:ff:ff in_port=1
```

## Task 2 (2)

3. Have a look at the OpenFlow 1.0 based program that implements the filtering switch described in the last slide: `simple_switch_filter.py`.
4. Create an OpenFlow 1.3 version called `simple_switch_filter_13.py`.
  1. Base your implementation on `simple_switch_13.py` and `simple_switch_filter.py`
  2. Use two flow tables for the OpenFlow 1.3 version
    1. Use the first flow table for matching input ports and source MAC address
    2. Use the second flow table for sending the packets out the correct port.
5. Describe and discuss the differences regarding the number of flow rules used
  1. Consider different topologies and number of hosts for the sake of discussion even though running them with the `simple_switch_filter.py` is not possible
  2. Mathematically describe an upper bound for the number of flow rules
    1. For the OpenFlow 1.0 version
    2. For the OpenFlow 1.3 version



## ❖ Looking for a Python IDE?

- Eclipse with PyDev plugin
  - <http://pydev.org/>
- PyCharm (free for Student for research purposes)  
<https://www.jetbrains.com/student/>

## ❖ Things to be aware of

- Ryu relies on a concurrent networking library called Eventlet (<http://eventlet.net>)
- Debugging Eventlet requires special support. PyCharm supports this, but it has to be enabled:
  - “Update: In recent versions of PyCharm, the option has moved to the main settings dialog. You can enable it under Settings | Python Debugger | Gevent compatible debugging.”
  - Help: <https://www.jetbrains.com/help/pycharm/2016.1/python-debugger.html>