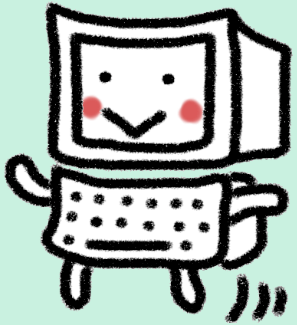


Project 3

ONOS Application Development:
SDN-enabled Learning Bridge

Deadline: 2020/10/21 (WED) 23:55

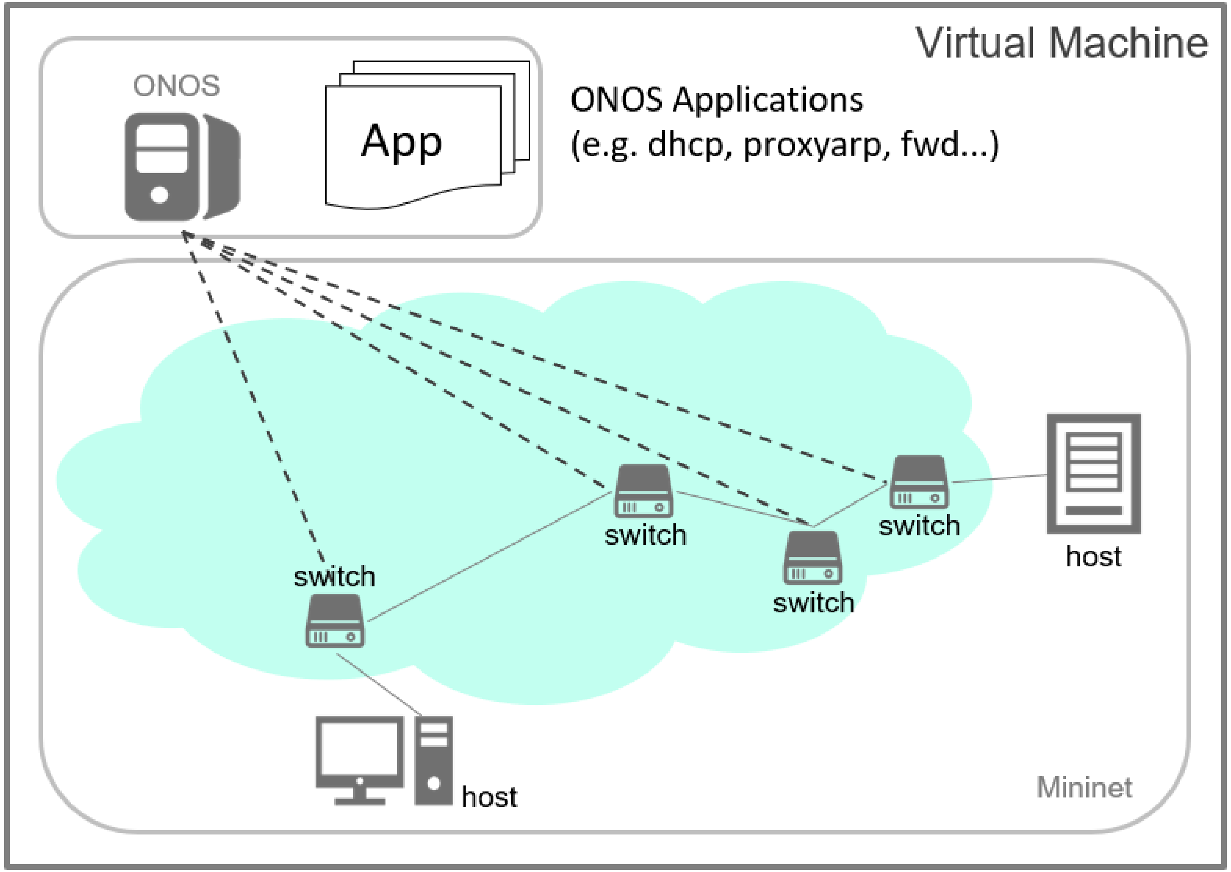
Outline

- Overview
- Build ONOS Application Project
  - Environment Setup
  - Create and Write ONOS Application
  - Compile, Install, and Activate ONOS Application
  - Reinstall ONOS Application
- Learning Bridge Function
  - Introduction
  - Workflow
- Project 3 Requirements
  - Create ONOS Application (10%)
  - Learning Bridge Function (60%)
  - Flow Rule Regulation (20%)
  - Submission Naming Convention (10%)
  - Restrictions

Outline

- ❑ **Overview**
- ❑ Build ONOS Application Project
  - ■ Environment Setup
  - ■ Create and Write ONOS Application
  - ■ Compile, Install, and Activate ONOS Application
  - ■ Reinstall ONOS Application
- ❑ Learning Bridge Function
  - ■ Introduction
  - ■ Workflow
- ❑ Project 3 Requirements
  - ■ Create ONOS Application (10%)
  - ■ Learning Bridge Function (60%)
  - ■ Flow Rule Regulation (20%)
  - ■ Submission Naming Convention (10%)
  - ■ Restrictions

Overview

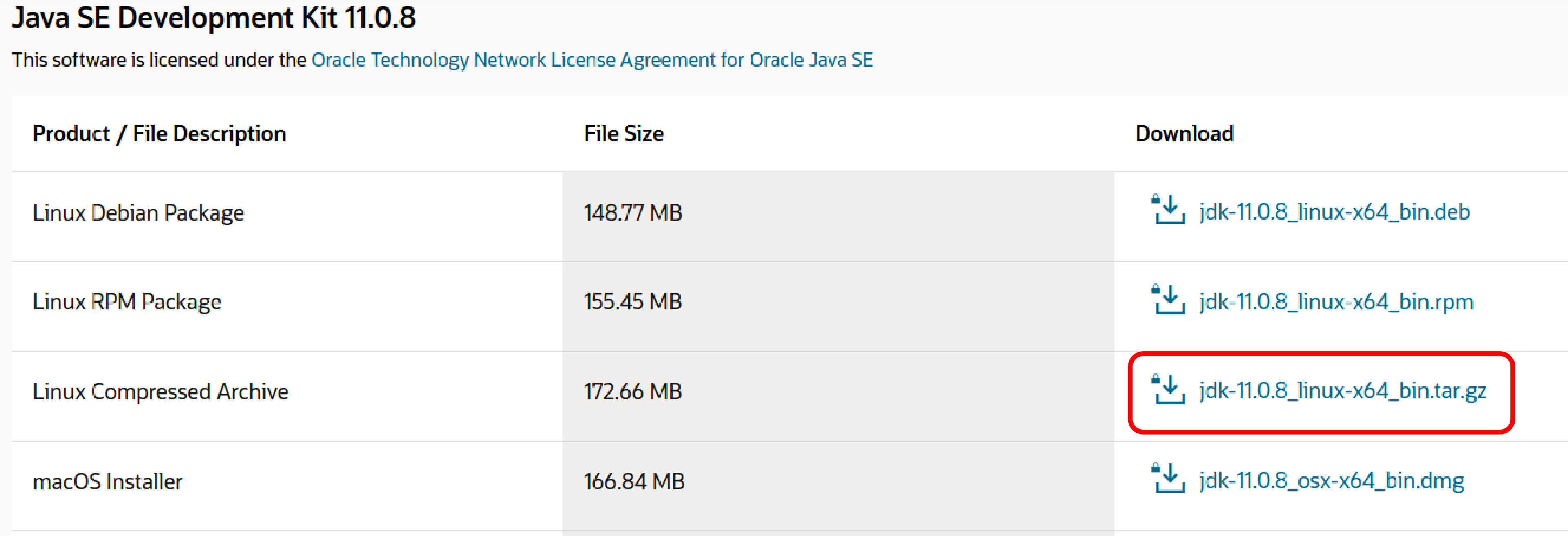

Virtual Machine
ONOS
App
ONOS Applications
(e.g. dhcp, proxyarp, fwd...)
switch
switch
switch
switch
host
host
Mininet

Outline

- Overview
- **Build ONOS Application Project**
  - **Environment Setup**
  - Create and Write ONOS Application
  - Compile, Install, and Activate ONOS Application
  - Reinstall ONOS Application
- Learning Bridge Function
  - Introduction
  - Workflow
- Project 3 Requirements
  - Create ONOS Application (10%)
  - Learning Bridge Function (60%)
  - Flow Rule Regulation (20%)
  - Submission Naming Convention (10%)
  - Restrictions

JDK installation (1/4)

- Download Oracle JDK 11 (JDK: Java Development Kit)
  - Java SE Development Kit 11- - Downloads



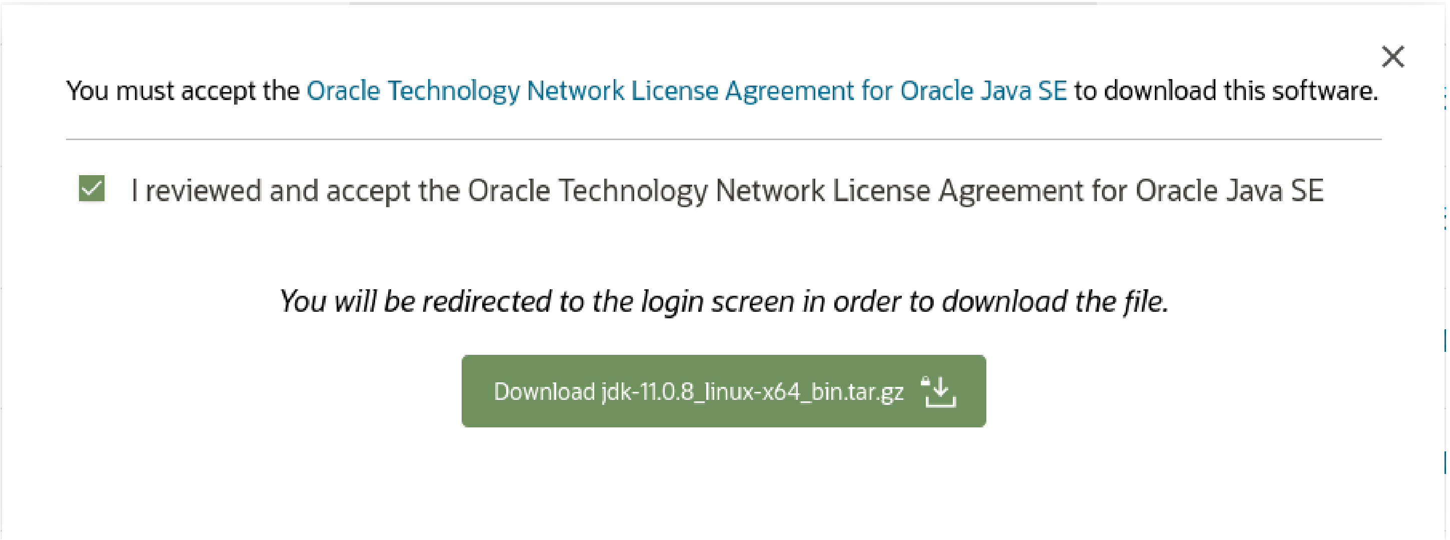
Java SE Development Kit 11.0.8

This software is licensed under the Oracle Technology Network License Agreement for Oracle Java SE

Product / File Description	File Size	Download
Linux Debian Package	148.77 MB	jdk-11.0.8_linux-x64_bin.deb
Linux RPM Package	155.45 MB	jdk-11.0.8_linux-x64_bin.rpm
Linux Compressed Archive	172.66 MB	jdk-11.0.8_linux-x64_bin.tar.gz
macOS Installer	166.84 MB	jdk-11.0.8_osx-x64_bin.dmg

JDK installation (2/4)

- Download Oracle JDK 11 (JDK: Java Development Kit)
  - You will be asked to create an Oracle account to download this software.



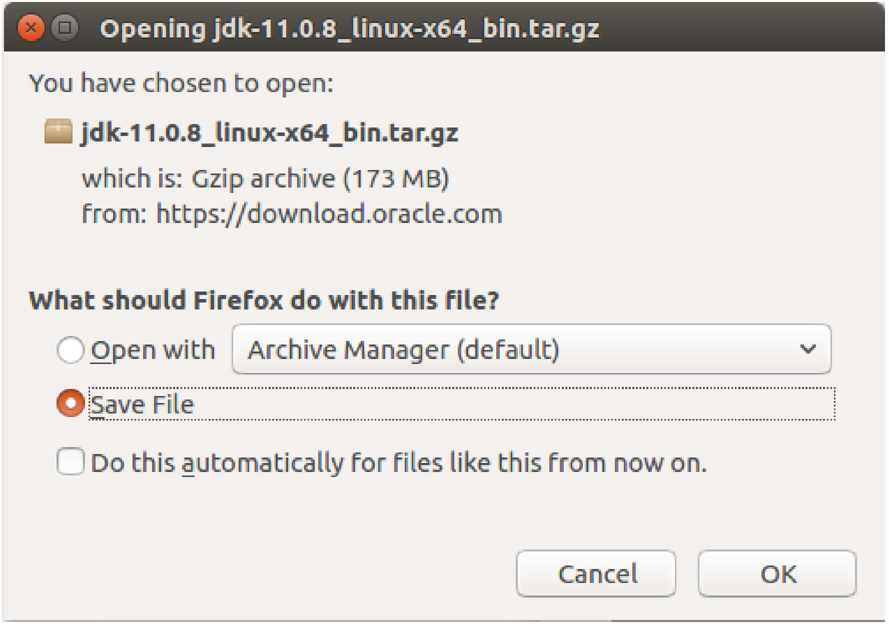
You must accept the Oracle Technology Network License Agreement for Oracle Java SE to download this software.

☑ I reviewed and accept the Oracle Technology Network License Agreement for Oracle Java SE

You will be redirected to the login screen in order to download the file.

JDK installation (3/4)

- Download Oracle JDK 11 (JDK: Java Development Kit)
  - After creating the Oracle account and login, you can download this file now.



Opening jdk-11.0.8_linux-x64_bin.tar.gz

You have chosen to open:

jdk-11.0.8_linux-x64_bin.tar.gz

which is: Gzip archive (173 MB)
from: https://download.oracle.com

What should Firefox do with this file?

○ Open with Archive Manager (default)

● Save File

□ Do this automatically for files like this from now on.

JDK installation (4/4)

- Untar JDK in */opt*

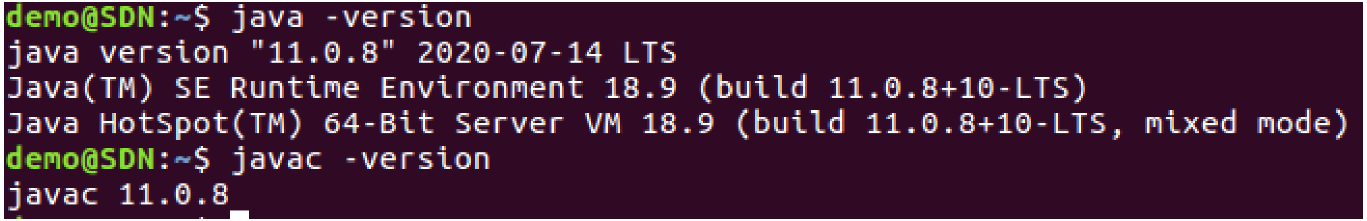
```
$ sudo tar –zxf ~/Downloads/jdk-11.0.8_linux-x64_bin.tar.gz –C /opt
```

- Set Oracle JDK as the default JVM

```
$ sudo update-alternatives --install /usr/bin/java java /opt/jdk-11.0.8/bin/java 2000
$ sudo update-alternatives --install /usr/bin/javac javac /opt/jdk-11.0.8/bin/javac 2000
```

- Check the result

```
$ java –version
$ javac -version
```



```
demo@SDN:~$ java -version
java version "11.0.8" 2020-07-14 LTS
Java(TM) SE Runtime Environment 18.9 (build 11.0.8+10-LTS)
Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.8+10-LTS, mixed mode)
demo@SDN:~$ javac -version
javac 11.0.8
```

Apache Maven installation

- ❑ Apache Maven
  - A software project management and comprehension tool
  - Based on the concept of a project object model (POM)
  - Can manage a project's build, reporting and documentation
- ❑ Install Maven

```
$ sudo apt install maven
```

- ❑ Indicate the version of ONOS API

```
$ export ONOS_POM_VERSION=2.2.0
```

- ❑ Build the current version of ONOS application archetypes
  - ■ ONOS version: 2.2.0

```
$ cd $ONOS_ROOT/tools/package/archetypes
$ mvn clean install -DskipTests
```

Outline

- ❑ Overview
- ❑ **Build ONOS Application Project**
  - ■ Environment Setup
  - ■ **Create and Write ONOS Application**
  - ■ Compile, Install, and Activate ONOS Application
  - ■ Reinstall ONOS Application
- ❑ Learning Bridge Function
  - ■ Introduction
  - ■ Workflow
- ❑ Project 3 Requirements
  - ■ Create ONOS Application (10%)
  - ■ Learning Bridge Function (60%)
  - ■ Flow Rule Regulation (20%)
  - ■ Submission Naming Convention (10%)
  - ■ Restrictions

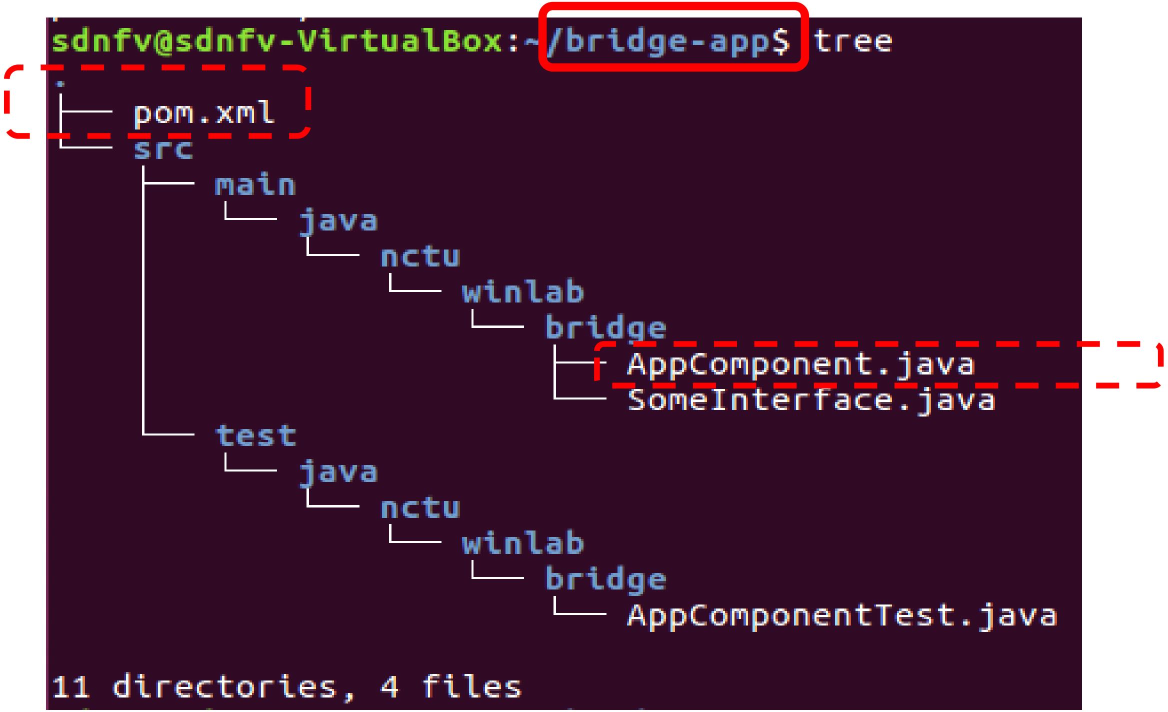
Create and Write ONOS Application (1/5)

- ❑ Create ONOS Application (Red words are what to type)

```
$ onos-create-app
…
[INFO] …
Define value for property 'groupId': nctu.winlab
Define value for property 'artifactId': bridge-app
Define value for property 'version' 1.0-SNAPSHOT: : <enter>
Define value for property 'package' nctu.winlab: : nctu.winlab.bridge
Confirm properties configuration:
groupId: nctu.winlab
artifactId: bridge-app
version: 1.0-SNAPSHOT
package: nctu.winlab.bridge
 Y: : <enter>
[INFO] …
…
[INFO] BUILD SUCCESS
```

Create and Write ONOS Application (2/5)

- After successful creation of application
  - ***onos-create-app*** creates a folder named <**artifactId**>.

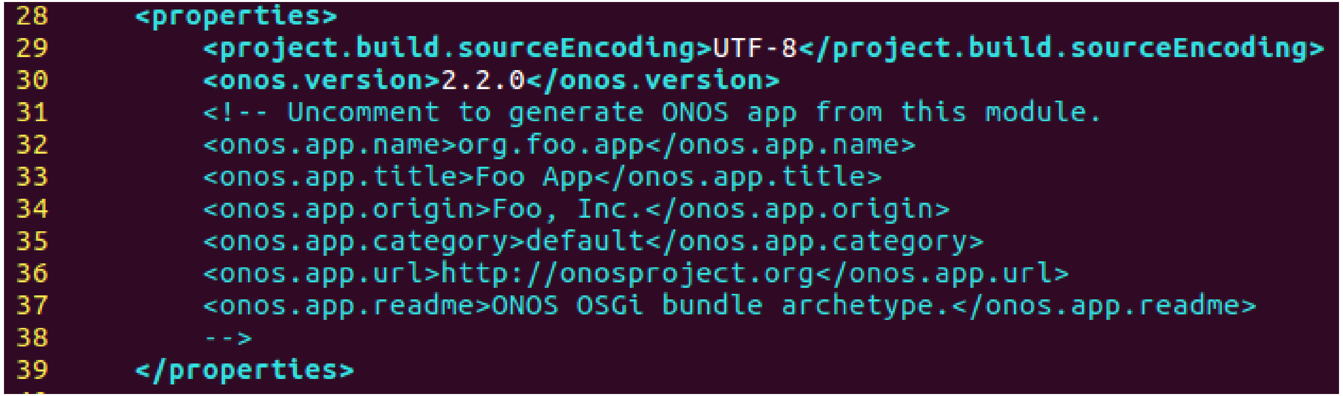


```
sdnfv@sdnfv-VirtualBox:~/bridge-app$ tree
.
├── pom.xml
└── src
    ├── main
    │   └── java
    │       └── nctu
    │           └── winlab
    │               └── bridge
    │                   ├── AppComponent.java
    │                   └── SomeInterface.java
    └── test
        └── java
            └── nctu
                └── winlab
                    └── bridge
                        └── AppComponentTest.java

11 directories, 4 files
```

Create and Write ONOS Application (3/5)

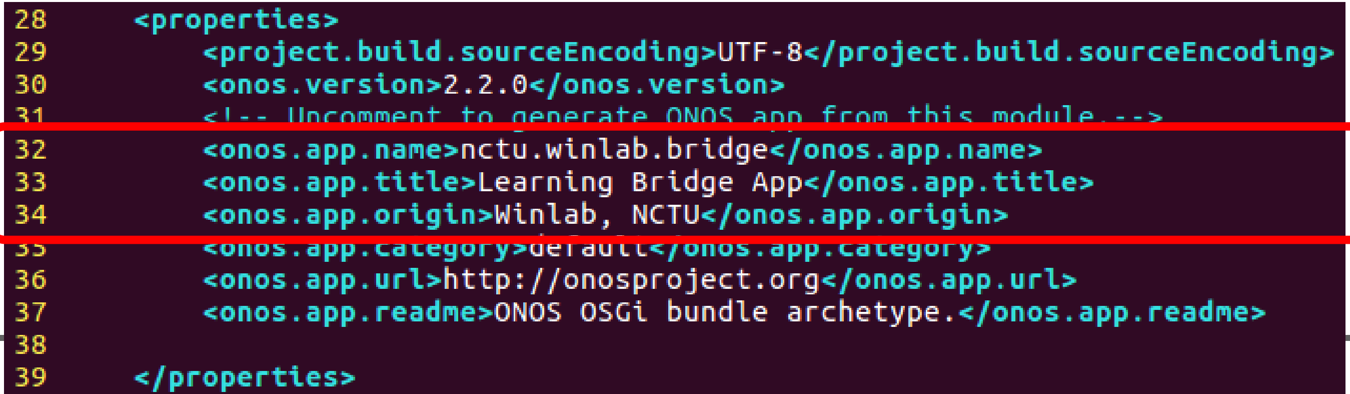
- Describe your project
  - By modifing **pom.xml** (pom: Project Object Model)

pom.xml
Before



```
    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <onos.version>2.2.0</onos.version>
        <!-- Uncomment to generate ONOS app from this module.
        <onos.app.name>org.foo.app</onos.app.name>
        <onos.app.title>Foo App</onos.app.title>
        <onos.app.origin>Foo, Inc.</onos.app.origin>
        <onos.app.category>default</onos.app.category>
        <onos.app.url>http://onosproject.org</onos.app.url>
        <onos.app.readme>ONOS OSGi bundle archetype.</onos.app.readme>
        -->
    </properties>
```

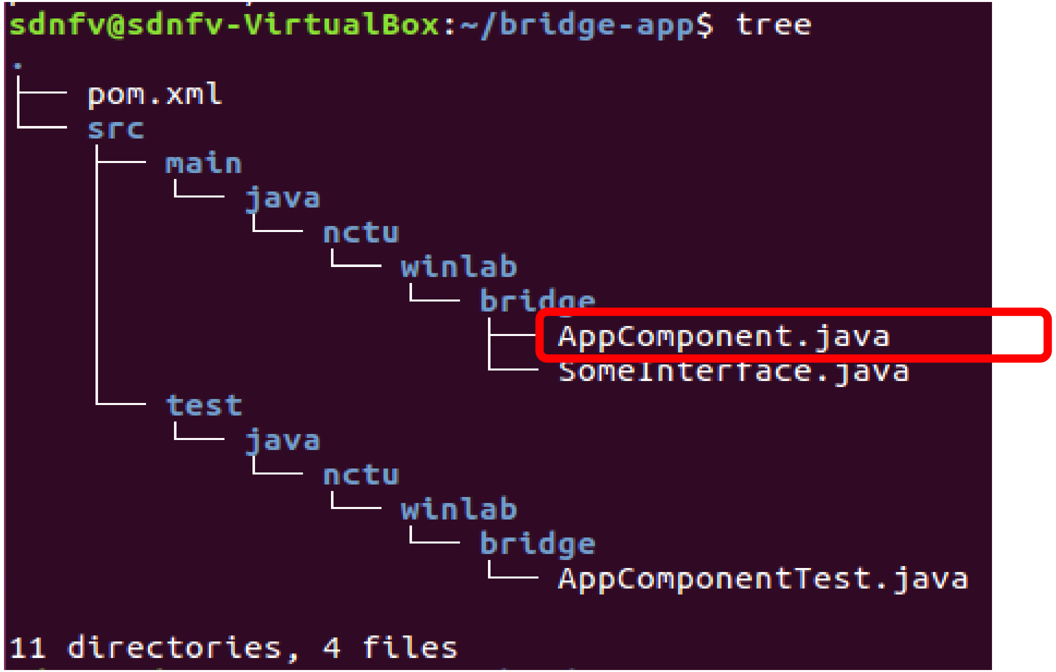
pom.xml
After



```
    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <onos.version>2.2.0</onos.version>
        <!-- Uncomment to generate ONOS app from this module.-->
        <onos.app.name>nctu.winlab.bridge</onos.app.name>
        <onos.app.title>Learning Bridge App</onos.app.title>
        <onos.app.origin>Winlab, NCTU</onos.app.origin>
        <onos.app.category>default</onos.app.category>
        <onos.app.url>http://onosproject.org</onos.app.url>
        <onos.app.readme>ONOS OSGi bundle archetype.</onos.app.readme>

    </properties>
```

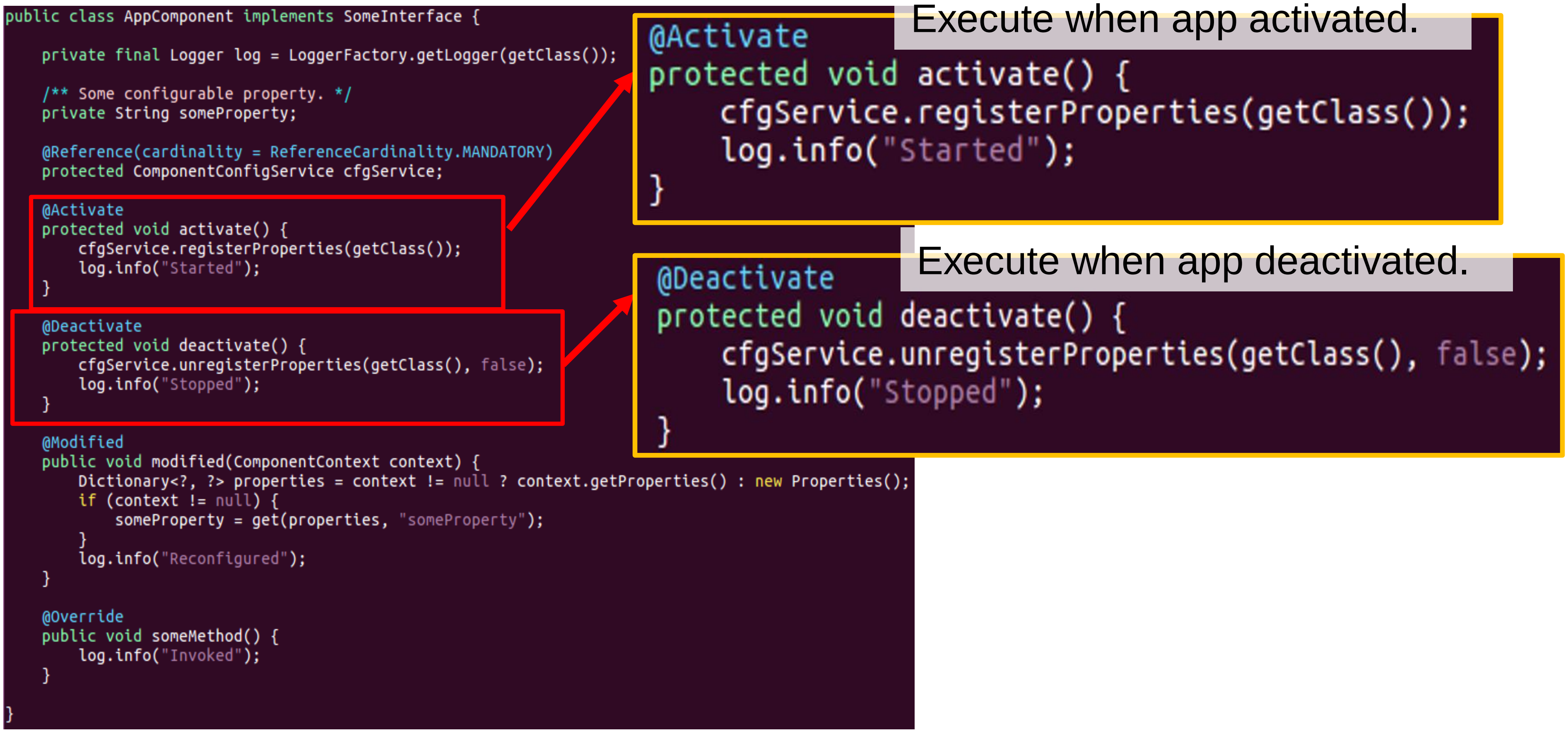
Create and Write ONOS Application (4/5)

- Find the template code in the application fold *`<artifactId>/src/main/java/nctu/winlab/bridge/`*



```
sdnfv@sdnfv-VirtualBox:~/bridge-app$ tree
.
├── pom.xml
└── src
    ├── main
    │   └── java
    │       └── nctu
    │           └── winlab
    │               └── bridge
    │                   ├── AppComponent.java
    │                   └── SomeInterface.java
    └── test
        └── java
            └── nctu
                └── winlab
                    └── bridge
                        └── AppComponentTest.java

11 directories, 4 files
```

Create and Write ONOS Application (5/5)



```
public class AppComponent implements SomeInterface {

    private final Logger log = LoggerFactory.getLogger(getClass());

    /** Some configurable property. */
    private String someProperty;

    @Reference(cardinality = ReferenceCardinality.MANDATORY)
    protected ComponentConfigService cfgService;

    @Activate
    protected void activate() {
        cfgService.registerProperties(getClass());
        log.info("Started");
    }

    @Deactivate
    protected void deactivate() {
        cfgService.unregisterProperties(getClass(), false);
        log.info("Stopped");
    }

    @Modified
    public void modified(ComponentContext context) {
        Dictionary<?, ?> properties = context != null ? context.getProperties() : new Properties();
        if (context != null) {
            someProperty = get(properties, "someProperty");
        }
        log.info("Reconfigured");
    }

    @Override
    public void someMethod() {
        log.info("Invoked");
    }

}
```

Execute when app activated.

```
@Activate
protected void activate() {
    cfgService.registerProperties(getClass());
    log.info("Started");
}
```

Execute when app deactivated.

```
@Deactivate
protected void deactivate() {
    cfgService.unregisterProperties(getClass(), false);
    log.info("Stopped");
}
```

Outline

- ❑ Overview
- ❑ **Build ONOS Application Project**
  - ■ Environment Setup
  - ■ Create and Write ONOS Application
  - ■ **Compile, Install, and Activate ONOS Application**
  - ■ Reinstall ONOS Application
- ❑ Learning Bridge Function
  - ■ Introduction
  - ■ Workflow
- ❑ Project 3 Requirements
  - ■ Create ONOS Application (10%)
  - ■ Learning Bridge Function (60%)
  - ■ Flow Rule Regulation (20%)
  - ■ Submission Naming Convention (10%)
  - ■ Restrictions

Compile, Install and Activate ONOS Application

□ Compile ONOS application

```
$ cd <artifactId>
$ mvn clean install –DsktipTests
          # option '-DskipTests' to skip running the tests for our project
```

□ Run ONOS

```
$ bazel run onos-local -- clean debug
```

□ Install and activate ONOS application

```
$ onos-app localhost install! target/<artifactId>-<version>.oar
```

- `*install*' with exclamation mark: activate the application immediately after the application being installed on ONOS.

Outline

- ❑ Overview
- ❑ **Build ONOS Application Project**
  - ■ Environment Setup
  - ■ Create and Write ONOS Application
  - ■ Compile, Install, and Activate ONOS Application
  - ■ **Reinstall ONOS Application**
- ❑ Learning Bridge Function
  - ■ Introduction
  - ■ Workflow
- ❑ Project 3 Requirements
  - ■ Create ONOS Application (10%)
  - ■ Learning Bridge Function (60%)
  - ■ Flow Rule Regulation (20%)
  - ■ Submission Naming Convention (10%)
  - ■ Restrictions

Reinstall ONOS Application

❑ Reinstall your application

- If you modify your application you need to recompile and reinstall your application on ONOS.

1. Recompile application by Maven

```
$ cd <artifactId> && mvn clean install -DskipTests
```

2. Deactivate application on ONOS

# <onos-app-name> is indicated in your pom.xml

```
$ onos localhost app deactivate <onos-app-name>
```

```
#e.g. nctu.winlab.bridge-app
```

3. Uninstall application

```
$ onos-app localhost uninstall <onos-app-name>
```

4. Install and Activate application

```
$ onos-app localhost install! target/<artifactId>-<version>.oar
```

References

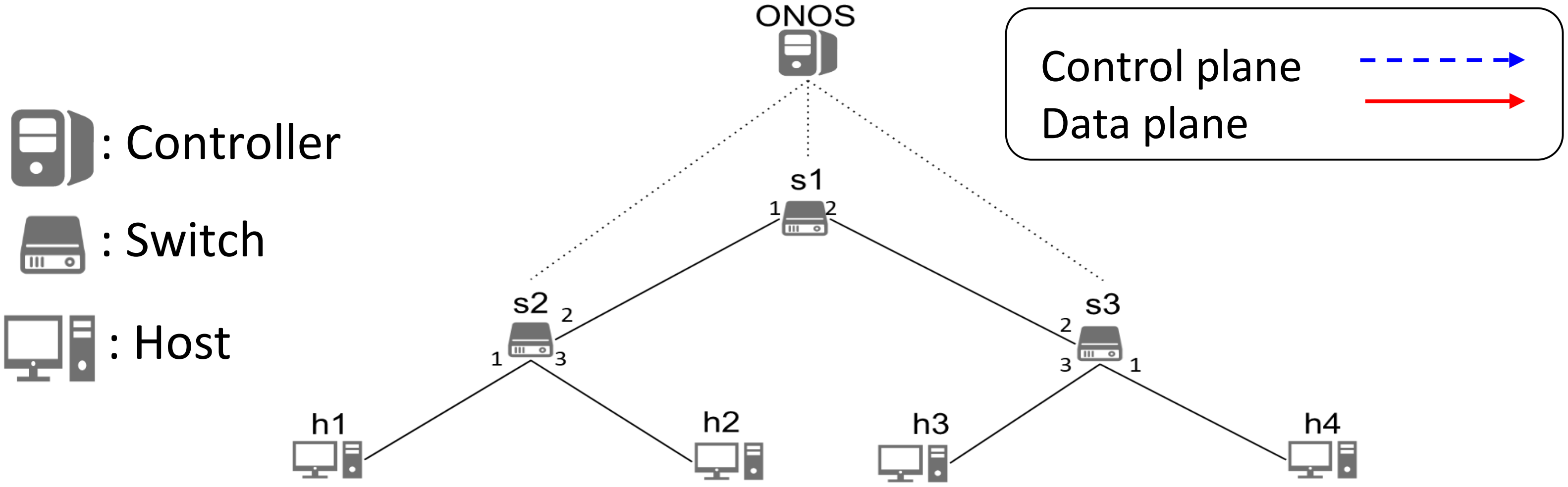
- ❑ ONOS Wiki – Template Application Tutorial
  - https://wiki.onosproject.org/display/ONOS/Template+Application+Tutorial
- ❑ ONOS Application Subsystem
  - https://wiki.onosproject.org/display/ONOS/Application+Subsystem
- ❑ ONOS Java API (2.2.0)
  - http://api.onosproject.org/2.2.0/apidocs/
- ❑ JDK installation
  - https://www.digitalocean.com/community/tutorials/how-to-manually-install-oracle-java-on-a-debian-or-ubuntu-vps

Outline

- ❑ Overview
- ❑ Build ONOS Application Project
  - ■ Environment Setup
  - ■ Create and Write ONOS Application
  - ■ Compile, Install, and Activate ONOS Application
  - ■ Reinstall ONOS Application
- ❑ **Learning Bridge Function**
  - ■ **Introduction**
  - ■ Workflow
- ❑ Project 3 Requirements
  - ■ Create ONOS Application (10%)
  - ■ Learning Bridge Function (60%)
  - ■ Flow Rule Regulation (20%)
  - ■ Submission Naming Convention (10%)
  - ■ Restrictions

Introduction of Learning Bridge Function

1. **Forwarding information learning**
   - Associate the source MAC address with incoming port
2. **Packets forwarding**
   - Use destination MAC address as index to look up the MAC address table and forward the packet to the proper output port



Control plane - - - -→
Data plane ——→

MAC=**00:00:00:00:00:01**

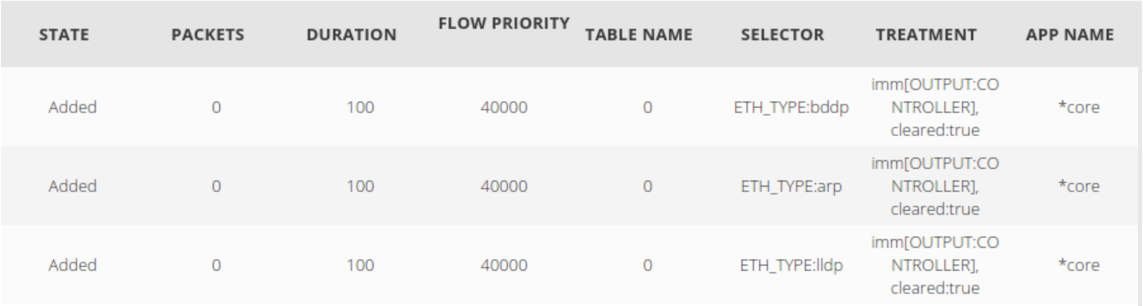
MAC=**00:00:00:00:00:04**

Outline

- ❑ Overview
- ❑ Build ONOS Application Project
  - ■ Environment Setup
  - ■ Create and Write ONOS Application
  - ■ Compile, Install, and Activate ONOS Application
  - ■ Reinstall ONOS Application
- ❑ **Learning Bridge Function**
  - ■ Introduction
  - ■ **Workflow**
- ❑ Project 3 Requirements
  - ■ Create ONOS Application (10%)
  - ■ Learning Bridge Function (60%)
  - ■ Flow Rule Regulation (20%)
  - ■ Submission Naming Convention (10%)
  - ■ Restrictions

Workflow of Learning Bridge Function

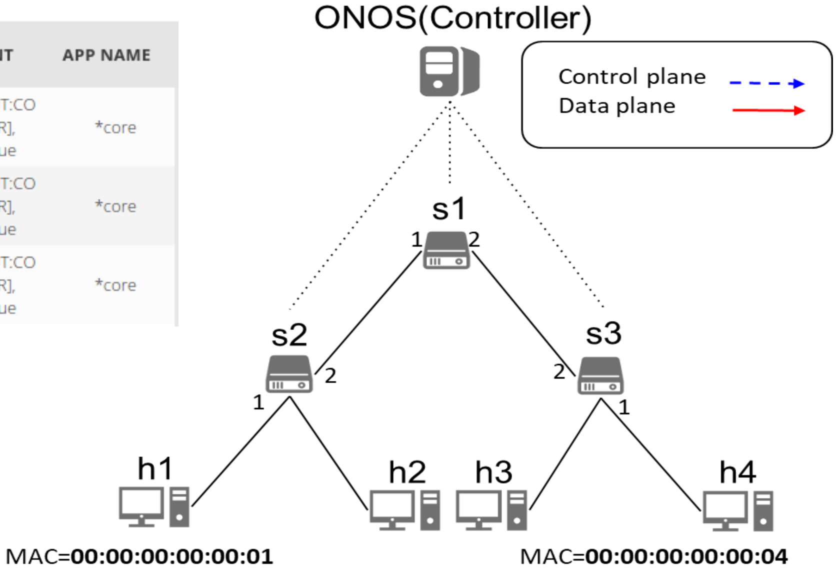
- ❑ Initially, MAC table and flow table are empty.
  - ■ Flow table (switch)



STATE	PACKETS	DURATION	FLOW PRIORITY	TABLE NAME	SELECTOR	TREATMENT	APP NAME
Added	0	100	40000	0	ETH_TYPE:bddp	imm[OUTPUT:CONTROLLER], cleared:true	*core
Added	0	100	40000	0	ETH_TYPE:arp	imm[OUTPUT:CONTROLLER], cleared:true	*core
Added	0	100	40000	0	ETH_TYPE:lldp	imm[OUTPUT:CONTROLLER], cleared:true	*core

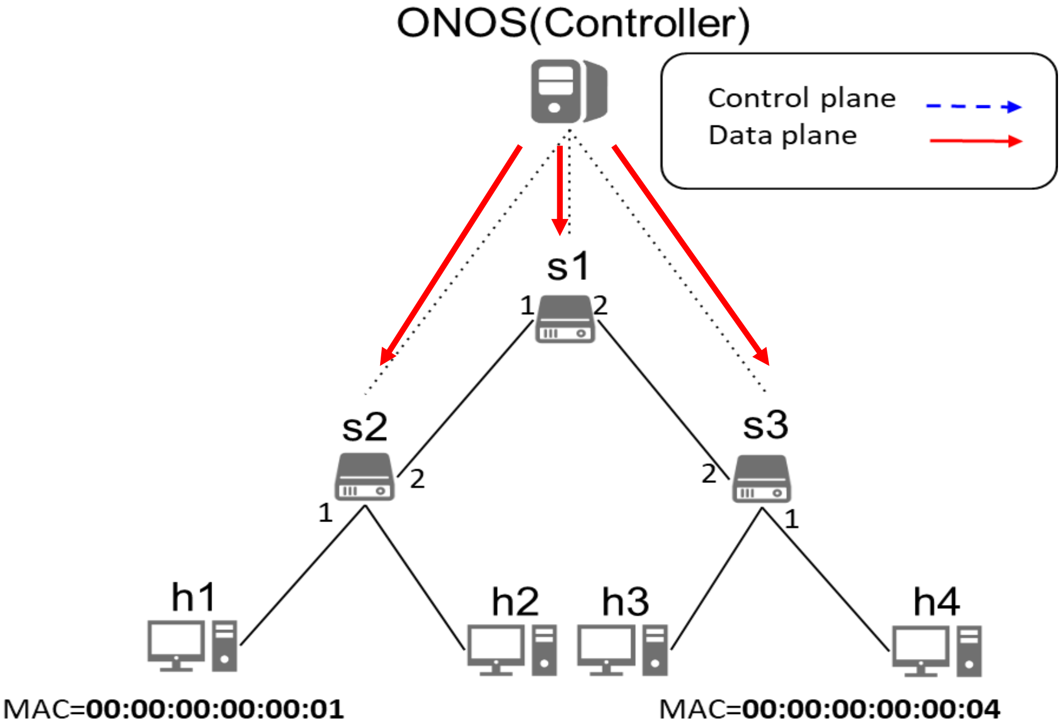
  - ■ MAC table (controller)

s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port



Workflow of Learning Bridge Function

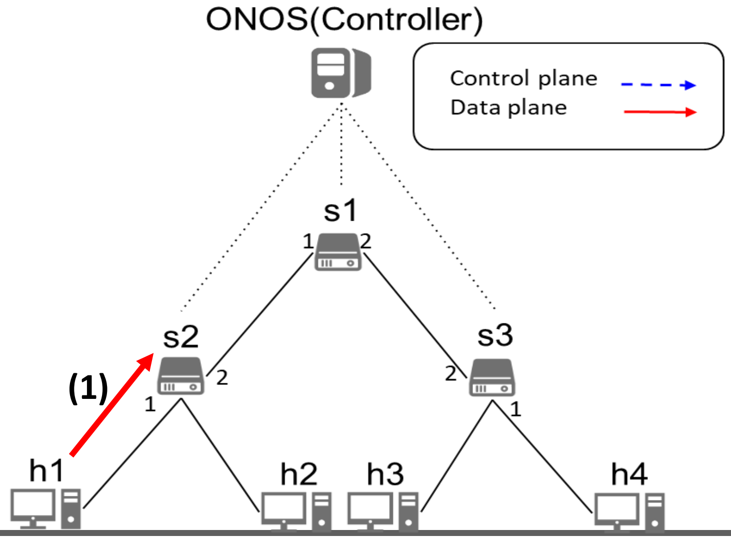
- When App is activated:
  - Install rules with very low priority to Packet-in on **ALL** switches.



Workflow (h1→h4)

1. h1 pings h4
2. Switch sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit**:
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h4 receives packet from h1

s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port



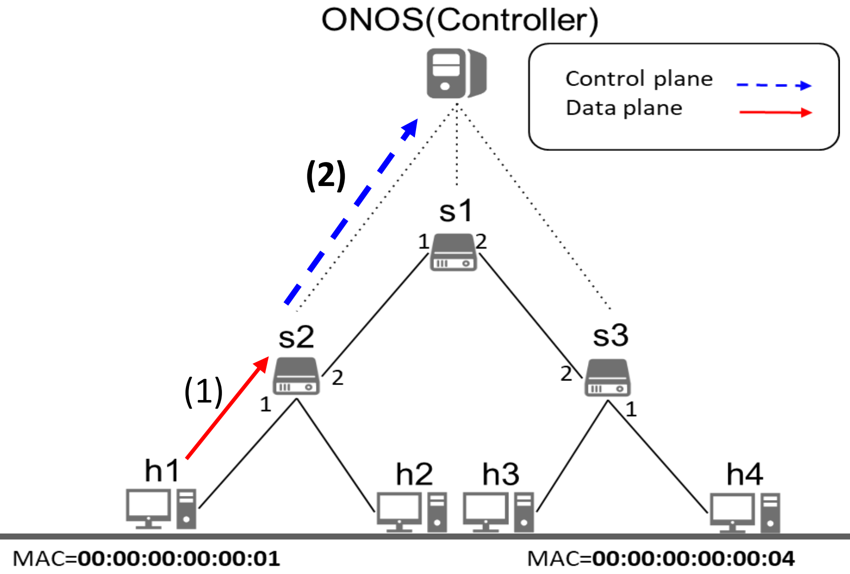
MAC=**00:00:00:00:00:01**

MAC=**00:00:00:00:00:04**

Workflow (h1→h4)

1. h1 pings h4
2. Switch (s2) sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss**:
      - Sends Packet out with flooding
   b. **Table hit:**
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h4 receives packet from h1

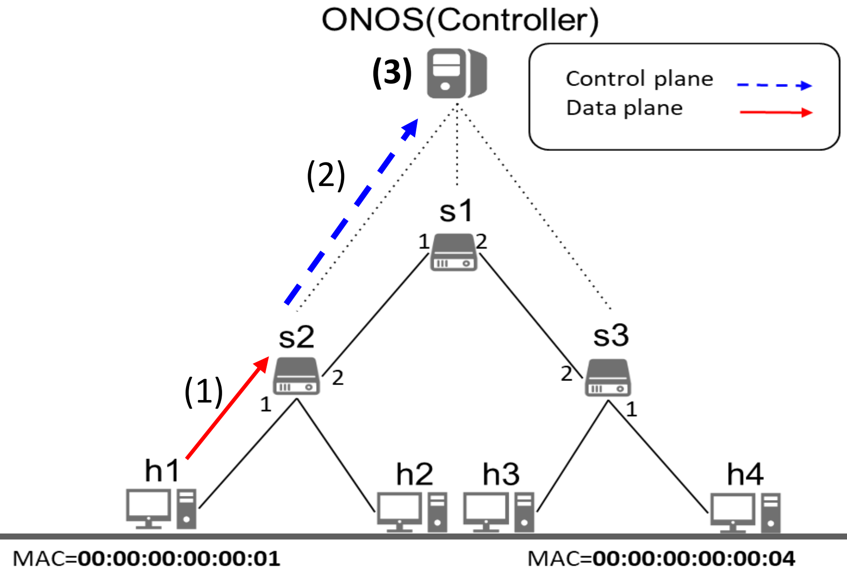
s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port



Workflow (h1→h4)

1. h1 pings h4
2. Switch sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit**:
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h4 receives packet from h1

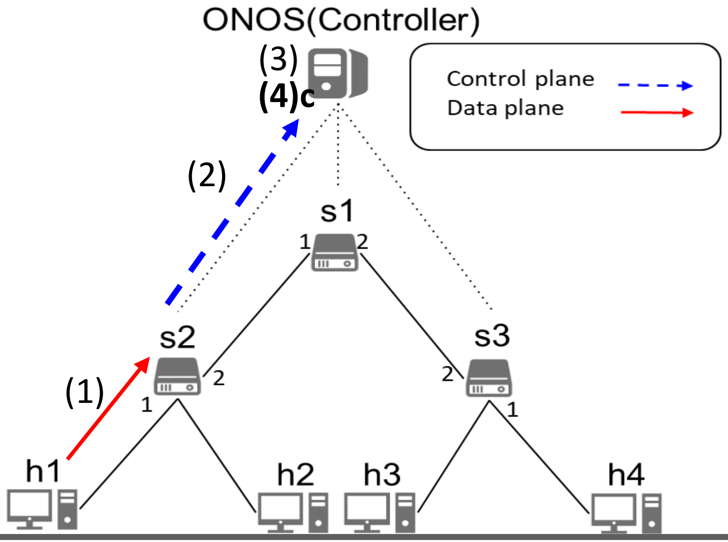
s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
		00:.....:01	**1**		



Workflow (h1→h4)

1. h1 pings h4
2. Switch sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit**:
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h4 receives packet from h1

s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
		00:......:01	1		



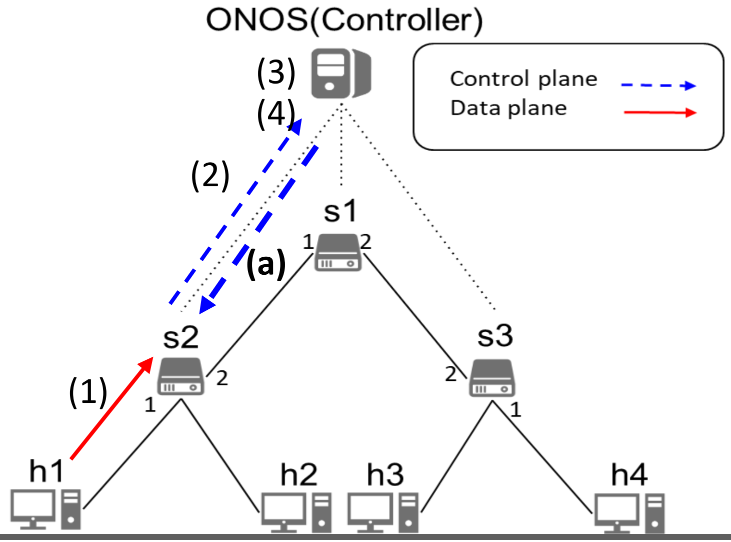
MAC=**00:00:00:00:00:01**

MAC=**00:00:00:00:00:04**

Workflow (h1→h4)

1. h1 pings h4
2. Switch sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit**:
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h4 receives packet from h1

s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
		00:......:01	1		



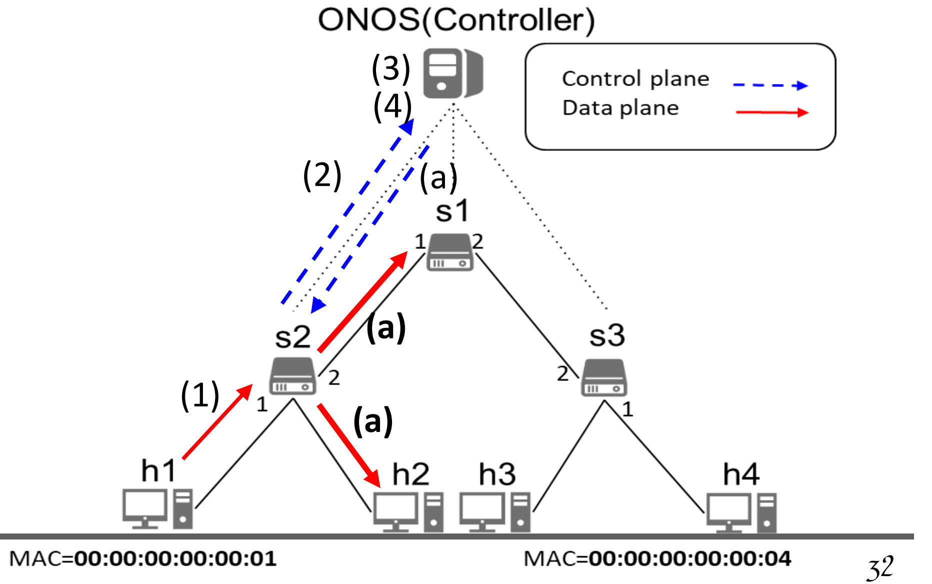
MAC=**00:00:00:00:00:01**

MAC=**00:00:00:00:00:04**

Workflow (h1→h4)

1. h1 pings h4
2. Switch sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      – Sends Packet out with flooding
   b. **Table hit**:
      – Sends Packet out with designated port
      – Installs flow rule on switch
5. h4 receives packet from h1

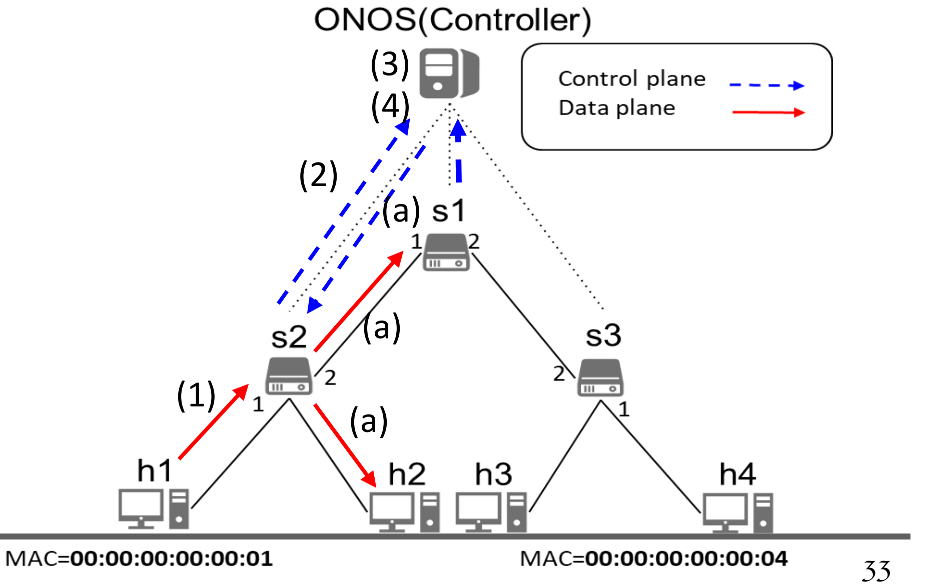
s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
		00:......:01	1		



Workflow (h1→h4)

1. h1 pings h4
2. Switch (s1) sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit**:
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h4 receives packet from h1

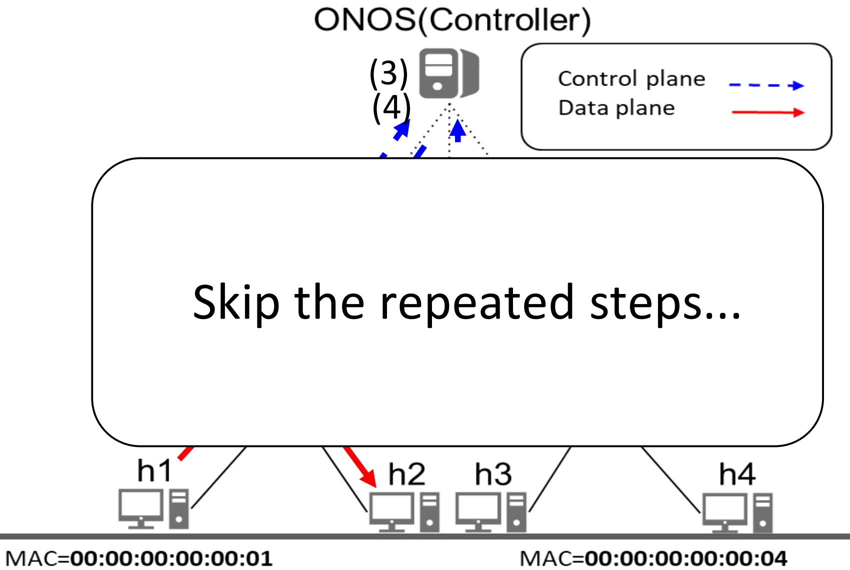
s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
		00:......:01	1		



Workflow (h1→h4)

1. h1 pings h4
2. Switch (s1) sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit:**
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h4 receives packet from h1

s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
		00:......:01	1		

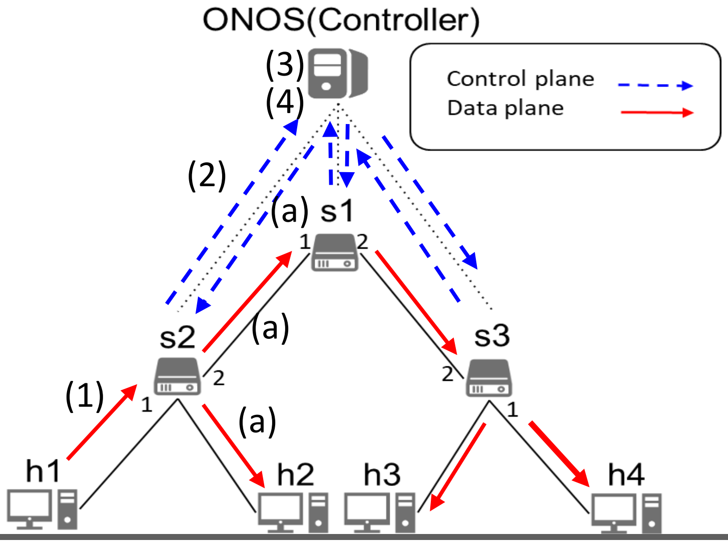


Skip the repeated steps...

Workflow (h1→h4)

1. h1 pings h4
2. Switch sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit**:
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h4 receives packet from h1

s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
00:......:01	1	00:......:01	1	00:......:01	2



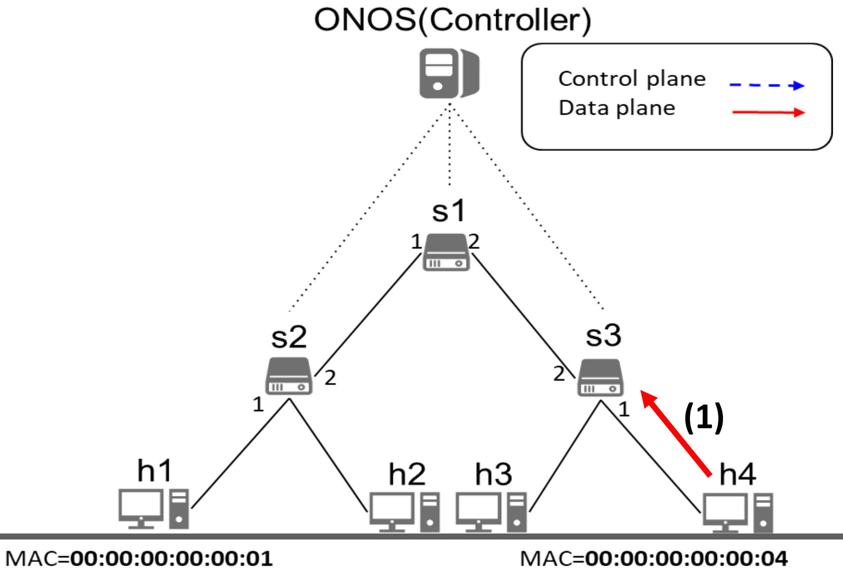
MAC=**00:00:00:00:00:01**

MAC=**00:00:00:00:00:04**

Workflow (h4→h1)

1. h4 replies to h1
2. Switch sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit**:
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h1 receives packet from h4

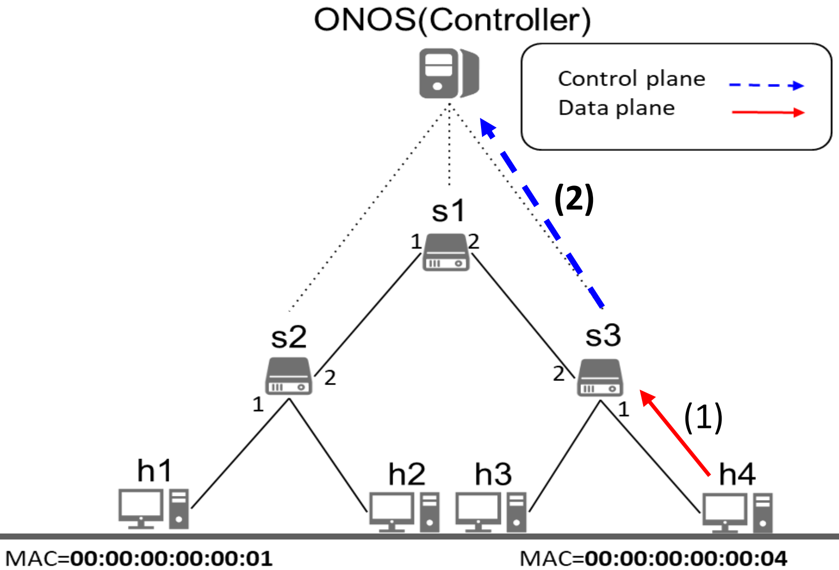
s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
00:......:01	1	00:......:01	1	00:......:01	2



Workflow (h4→h1)

1. h4 replies to h1
2. Switch (s3) sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit**:
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h1 receives packet from h4

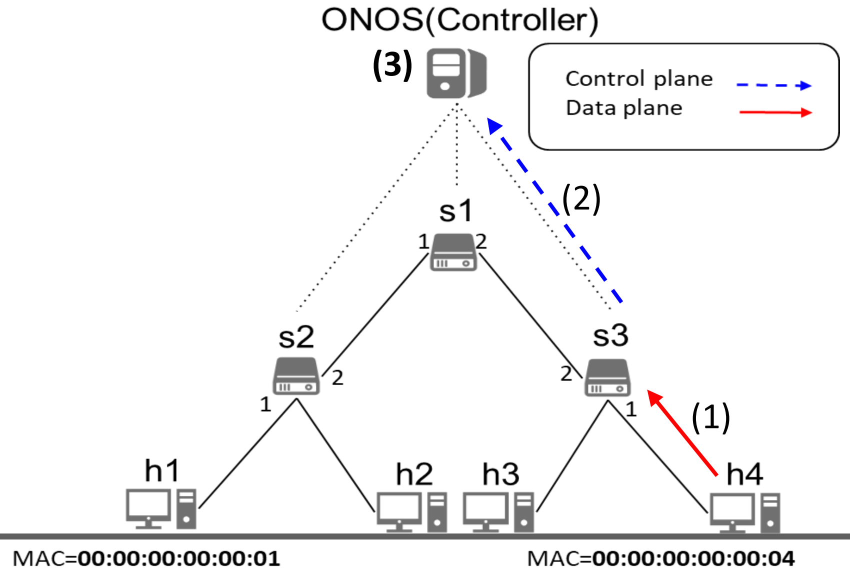
s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
00:......:01	1	00:......:01	1	00:......:01	2



Workflow (h4→h1)

1. h4 replies to h1
2. Switch sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit**:
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h1 receives packet from h4

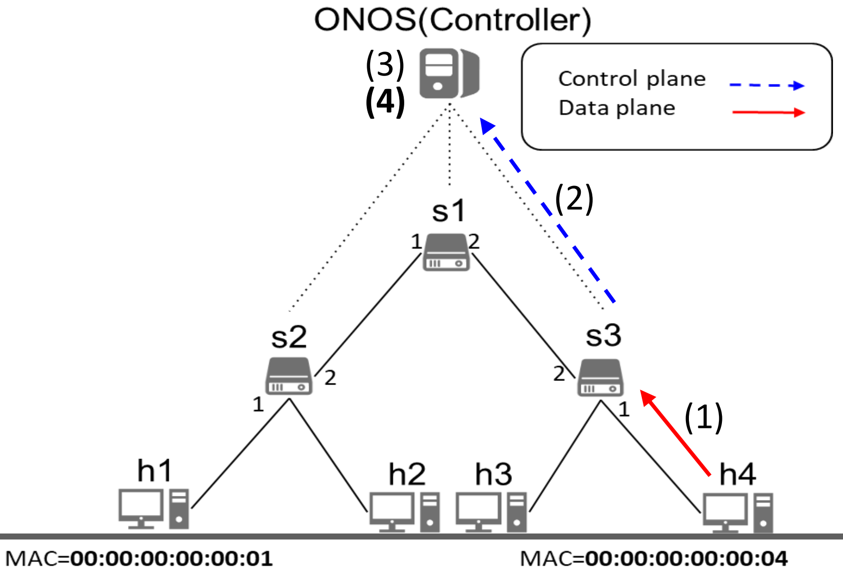
s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
00:......:01	1	00:......:01	1	00:......:01	2
				00:......:04	**1**



Workflow (h4→h1)

1. h4 replies to h1
2. Switch sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit**:
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h1 receives packet from h4

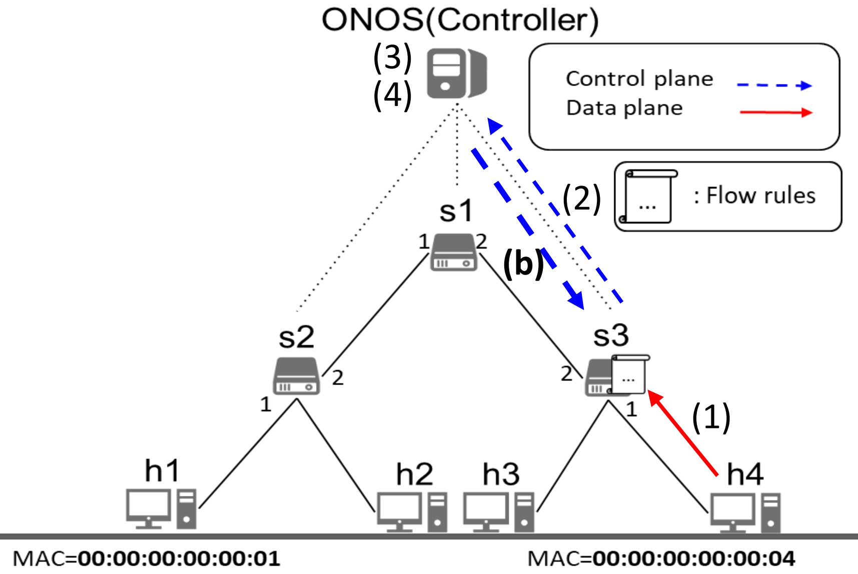
s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
00:......:01	1	00:......:01	1	00:......:01	2
				00:......:04	1



Workflow (h4→h1)

1. h4 replies to h1
2. Switch sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit**:
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h1 receives packet from h4

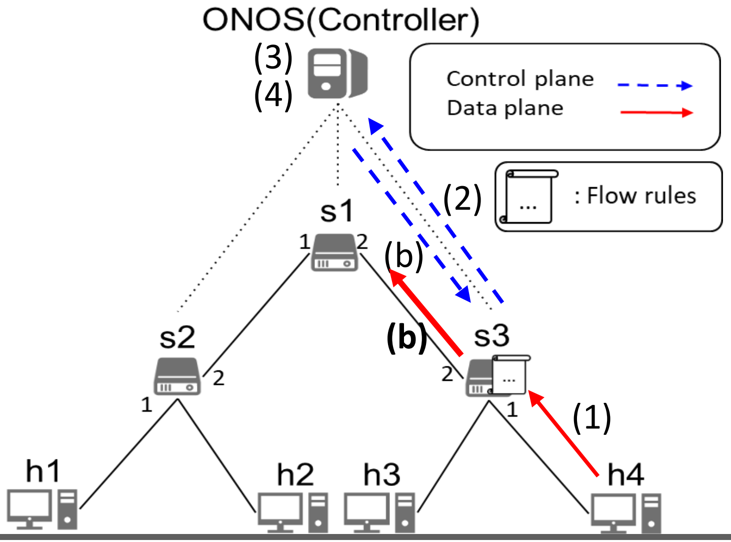
s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
00:......:01	1	00:......:01	1	00:......:01	2
				00:......:04	1



Workflow (h4→h1)

1. h4 replies to h1
2. Switch sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit**:
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h1 receives packet from h4

s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
00:......:01	1	00:......:01	1	00:......:01	2
				00:......:04	1



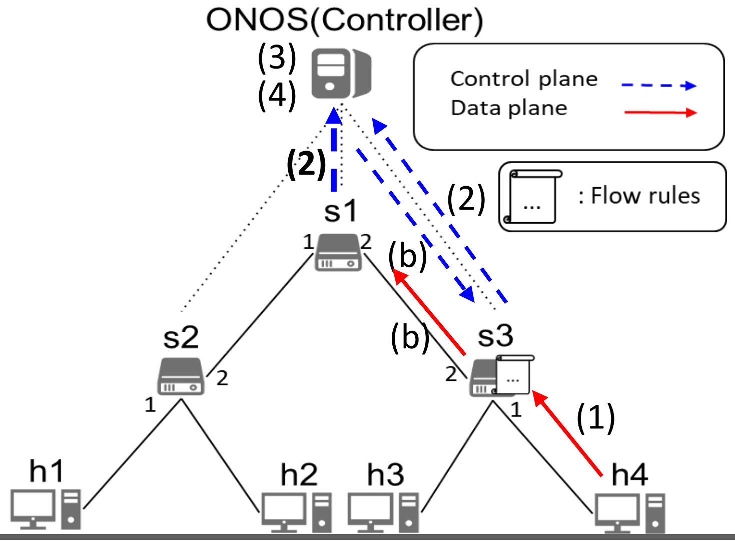
MAC=**00:00:00:00:00:01**

MAC=**00:00:00:00:00:04**

Workflow (h4→h1)

1. h4 replies to h1
2. Switch (s1) sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
    a. **Table miss:**
        - Sends Packet out with flooding
    b. **Table hit**:
        - Sends Packet out with designated port
        - Installs flow rule on switch
5. h1 receives packet from h4

s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
00:……:01	1	00:……:01	1	00:……:01	2
				00:……:04	1



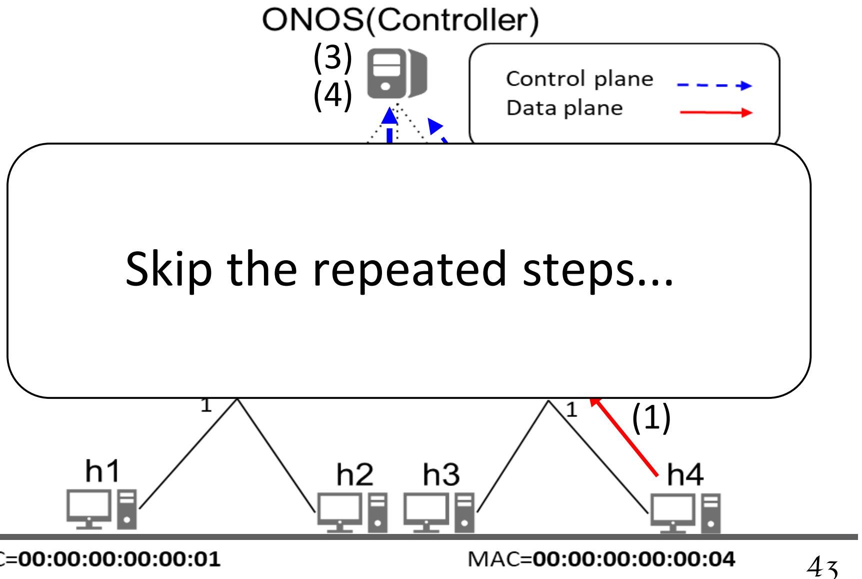
MAC=**00:00:00:00:00:01**

MAC=**00:00:00:00:00:04**

Workflow (h4→h1)

1. h4 replies to h1
2. Switch (s1) sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit**:
      - Sends Packet-out with designated port
      - Installs flow rule on switch
5. h1 receives packet from h4

s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
00:......:01	1	00:......:01	1	00:......:01	2
				00:......:04	1

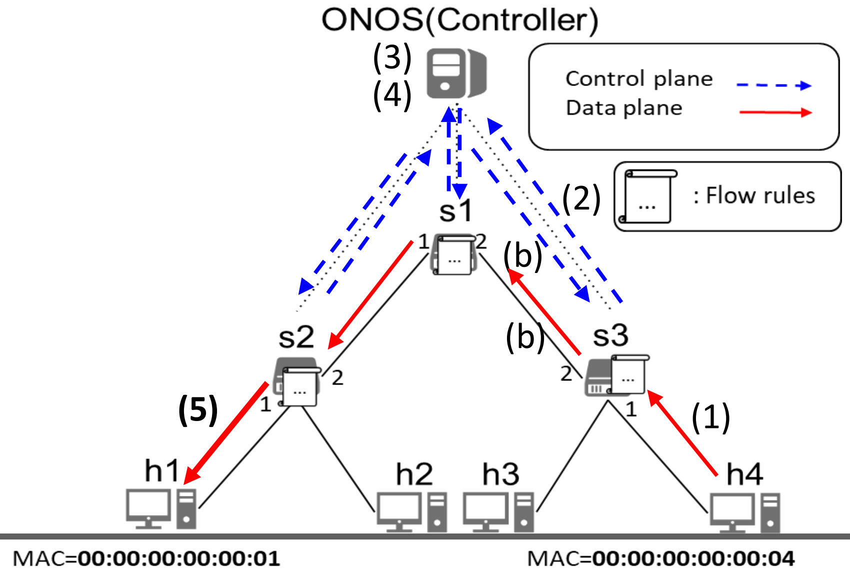


Skip the repeated steps...

Workflow (h4→h1)

1. h4 replies to h1
2. Switch sends Packet-in to Controller
3. Controller updates MAC address table with source MAC
4. Controller looks up MAC address table for destination MAC:
   a. **Table miss:**
      - Sends Packet out with flooding
   b. **Table hit**:
      - Sends Packet out with designated port
      - Installs flow rule on switch
5. h1 receives packet from h4

s1		s2		s3	
MAC	Port	MAC	Port	MAC	Port
00:......:01	1	00:......:01	1	00:......:01	2
00:......:04	2	00:......:04	2	00:......:04	1

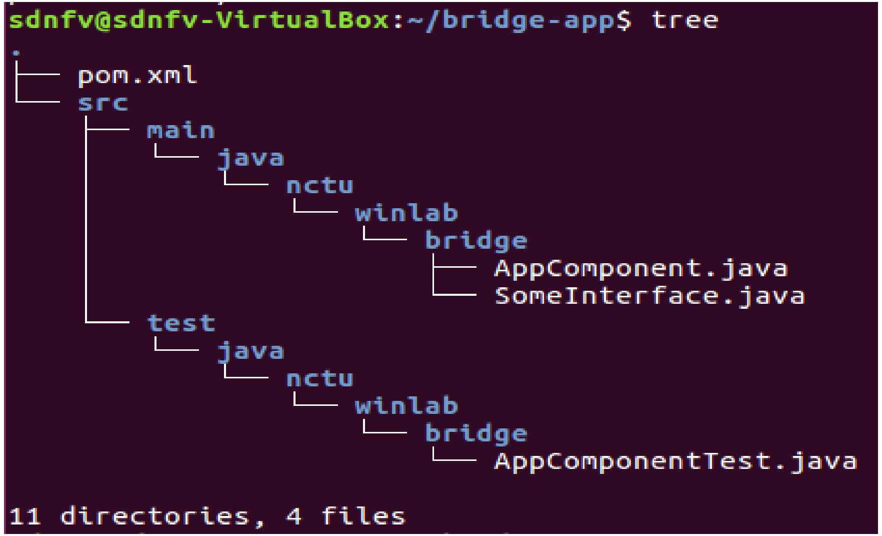


Outline

- ❑ Overview
- ❑ Build ONOS Application Project
  - ■ Environment Setup
  - ■ Create and Write ONOS Application
  - ■ Compile, Install, and Activate ONOS Application
  - ■ Reinstall ONOS Application
- ❑ Learning Bridge Function
  - ■ Introduction
  - ■ Workflow
- ❑ **Project3 Requirement**
  - ■ **Create ONOS Application (10%)**
  - ■ Learning Bridge Function (60%)
  - ■ Flow Rule Regulation (20%)
  - ■ Submission Naming Convention (10%)
  - ■ Restrictions

Create an ONOS application

- Maven project naming convention
  - Incorrect naming convention or format subjects to not scoring
    - <groupId>: nctu.winlab
    - <artifactId>: bridge-app
    - <version>: (default)
    - <package>: nctu.winlab.bridge



```
sdnfv@sdnfv-VirtualBox:~/bridge-app$ tree
.
├── pom.xml
└── src
    ├── main
    │   └── java
    │       └── nctu
    │           └── winlab
    │               └── bridge
    │                   ├── AppComponent.java
    │                   └── SomeInterface.java
    └── test
        └── java
            └── nctu
                └── winlab
                    └── bridge
                        └── AppComponentTest.java

11 directories, 4 files
```

Outline

- □ Overview
- □ Build ONOS Application Project
  - ■ Environment Setup
  - ■ Create and Write ONOS Application
  - ■ Compile, Install, and Activate ONOS Application
  - ■ Reinstall ONOS Application
- □ Learning Bridge Function
  - ■ Introduction
  - ■ Workflow
- □ **Project3 Requirement**
  - ■ Create ONOS Application (10%)
  - ■ **Learning Bridge Function (60%)**
  - ■ Flow Rule Regulation (20%)
  - ■ Submission Naming Convention (10%)
  - ■ Restrictions

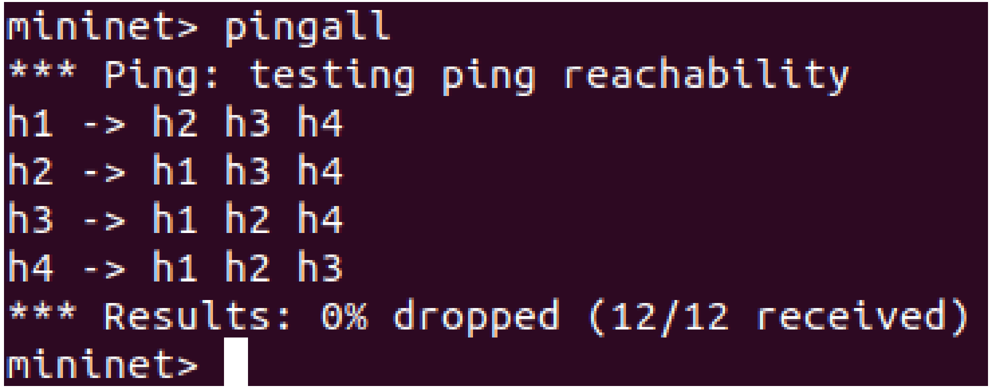
Learning Bridge Function

- ❑ **Learning Bridge & Forwarding Packet**
  a. Learning Bridge Function with ***tree*** (depth=2) topology **(20%)**
  b. Learning Bridge Function with ***tree*** (depth=3~5) topology **(20%)**

```
$ sudo mn --controller=remote,127.0.0.1:6653 --topo=tree,depth=2
```

- ❑ Ping should work between all hosts.

```
mininet> pingall
```

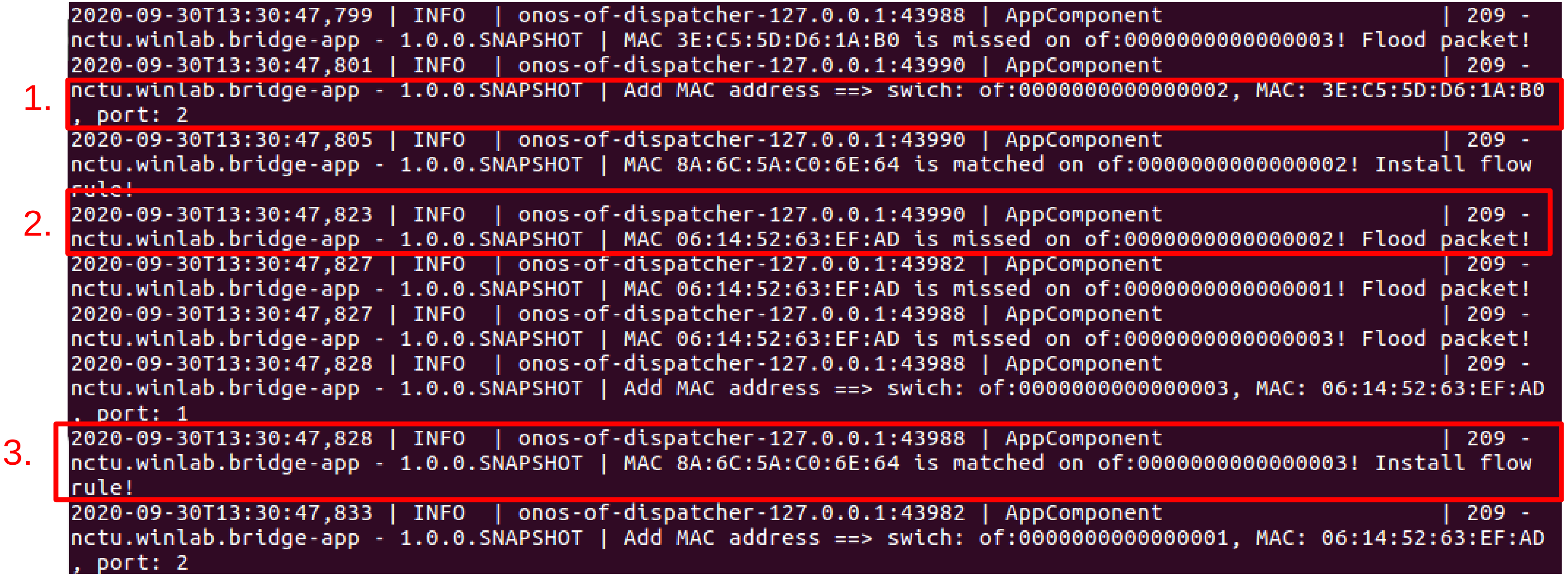


```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
mininet>
```

(Ex. mininet ***tree*** topology with depth=2)

Learning Bridge Function

- ❑ Use “log.info()” to print the status of learning bridge table . **(20%)**
  1. New MAC address added into the table.
  2. Table miss, packet flooded.
  3. Table hit, flow rule installed on the switch.



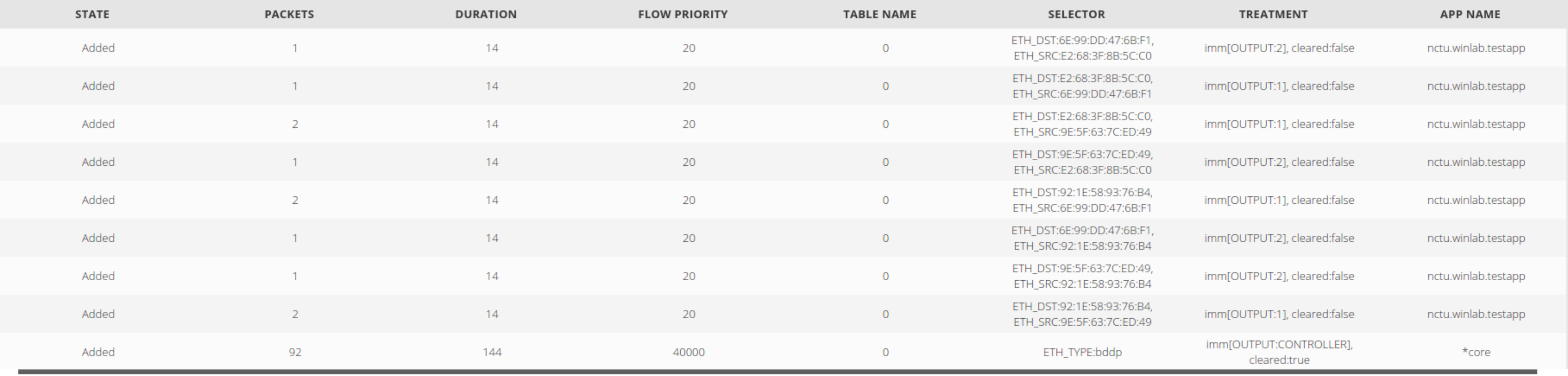
```
2020-09-30T13:30:47,799 | INFO  | onos-of-dispatcher-127.0.0.1:43988 | AppComponent                     | 209 -
nctu.winlab.bridge-app - 1.0.0.SNAPSHOT | MAC 3E:C5:5D:D6:1A:B0 is missed on of:0000000000000003! Flood packet!
2020-09-30T13:30:47,801 | INFO  | onos-of-dispatcher-127.0.0.1:43990 | AppComponent                     | 209 -
nctu.winlab.bridge-app - 1.0.0.SNAPSHOT | Add MAC address ==> swich: of:0000000000000002, MAC: 3E:C5:5D:D6:1A:B0
, port: 2
2020-09-30T13:30:47,805 | INFO  | onos-of-dispatcher-127.0.0.1:43990 | AppComponent                     | 209 -
nctu.winlab.bridge-app - 1.0.0.SNAPSHOT | MAC 8A:6C:5A:C0:6E:64 is matched on of:0000000000000002! Install flow
rule!
2020-09-30T13:30:47,823 | INFO  | onos-of-dispatcher-127.0.0.1:43990 | AppComponent                     | 209 -
nctu.winlab.bridge-app - 1.0.0.SNAPSHOT | MAC 06:14:52:63:EF:AD is missed on of:0000000000000002! Flood packet!
2020-09-30T13:30:47,827 | INFO  | onos-of-dispatcher-127.0.0.1:43982 | AppComponent                     | 209 -
nctu.winlab.bridge-app - 1.0.0.SNAPSHOT | MAC 06:14:52:63:EF:AD is missed on of:0000000000000001! Flood packet!
2020-09-30T13:30:47,827 | INFO  | onos-of-dispatcher-127.0.0.1:43988 | AppComponent                     | 209 -
nctu.winlab.bridge-app - 1.0.0.SNAPSHOT | MAC 06:14:52:63:EF:AD is missed on of:0000000000000003! Flood packet!
2020-09-30T13:30:47,828 | INFO  | onos-of-dispatcher-127.0.0.1:43988 | AppComponent                     | 209 -
nctu.winlab.bridge-app - 1.0.0.SNAPSHOT | Add MAC address ==> swich: of:0000000000000003, MAC: 06:14:52:63:EF:AD
, port: 1
2020-09-30T13:30:47,828 | INFO  | onos-of-dispatcher-127.0.0.1:43988 | AppComponent                     | 209 -
nctu.winlab.bridge-app - 1.0.0.SNAPSHOT | MAC 8A:6C:5A:C0:6E:64 is matched on of:0000000000000003! Install flow
rule!
2020-09-30T13:30:47,833 | INFO  | onos-of-dispatcher-127.0.0.1:43982 | AppComponent                     | 209 -
nctu.winlab.bridge-app - 1.0.0.SNAPSHOT | Add MAC address ==> swich: of:0000000000000001, MAC: 06:14:52:63:EF:AD
, port: 2
```

Outline

- ❑ Overview
- ❑ Build ONOS Application Project
  - ■ Environment Setup
  - ■ Create and Write ONOS Application
  - ■ Compile, Install, and Activate ONOS Application
  - ■ Reinstall ONOS Application
- ❑ Learning Bridge Function
  - ■ Introduction
  - ■ Workflow
- ❑ **Project3 Requirement**
  - ■ Create ONOS Application (10%)
  - ■ Learning Bridge Function (60%)
  - ■ **Flow Rule Regulation (20%)**
  - ■ Submission Naming Convention (10%)
  - ■ Restrictions

Flow Rule Regulation

- Rule Requirement **(20%)**
  - Match field (selector): ETH_SRC, ETH_DST
  - Action field (treatment): OUTPUT
  - Flow priority: 20
  - Flow timeout: 20



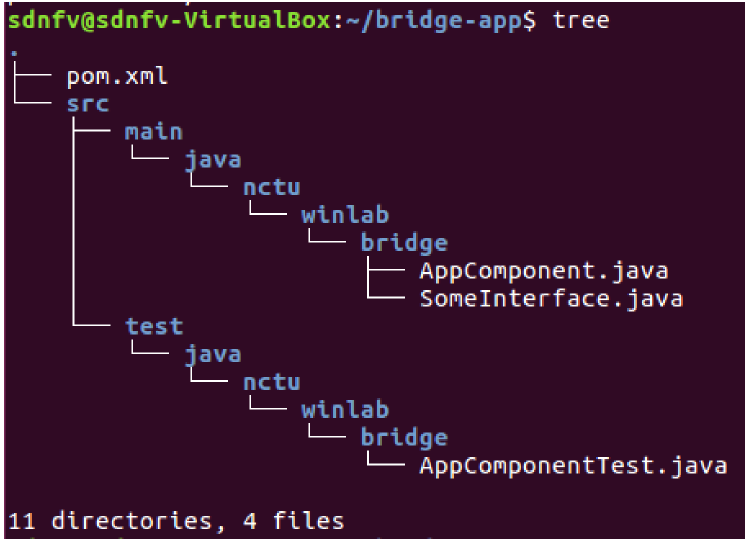
STATE	PACKETS	DURATION	FLOW PRIORITY	TABLE NAME	SELECTOR	TREATMENT	APP NAME
Added	1	14	20	0	ETH_DST:6E:99:DD:47:6B:F1, ETH_SRC:E2:68:3F:8B:5C:C0	imm[OUTPUT:2], cleared:false	nctu.winlab.testapp
Added	1	14	20	0	ETH_DST:E2:68:3F:8B:5C:C0, ETH_SRC:6E:99:DD:47:6B:F1	imm[OUTPUT:1], cleared:false	nctu.winlab.testapp
Added	2	14	20	0	ETH_DST:E2:68:3F:8B:5C:C0, ETH_SRC:9E:5F:63:7C:ED:49	imm[OUTPUT:1], cleared:false	nctu.winlab.testapp
Added	1	14	20	0	ETH_DST:9E:5F:63:7C:ED:49, ETH_SRC:E2:68:3F:8B:5C:C0	imm[OUTPUT:2], cleared:false	nctu.winlab.testapp
Added	2	14	20	0	ETH_DST:92:1E:58:93:76:B4, ETH_SRC:6E:99:DD:47:6B:F1	imm[OUTPUT:1], cleared:false	nctu.winlab.testapp
Added	1	14	20	0	ETH_DST:6E:99:DD:47:6B:F1, ETH_SRC:92:1E:58:93:76:B4	imm[OUTPUT:2], cleared:false	nctu.winlab.testapp
Added	1	14	20	0	ETH_DST:9E:5F:63:7C:ED:49, ETH_SRC:92:1E:58:93:76:B4	imm[OUTPUT:2], cleared:false	nctu.winlab.testapp
Added	2	14	20	0	ETH_DST:92:1E:58:93:76:B4, ETH_SRC:9E:5F:63:7C:ED:49	imm[OUTPUT:1], cleared:false	nctu.winlab.testapp
Added	92	144	40000	0	ETH_TYPE:bddp	imm[OUTPUT:CONTROLLER], cleared:true	*core

Outline

- □ Overview
- □ Build ONOS Application Project
  - ■ Environment Setup
  - ■ Create and Write ONOS Application
  - ■ Compile, Install, and Activate ONOS Application
  - ■ Reinstall ONOS Application
- □ Learning Bridge Function
  - ■ Introduction
  - ■ Workflow
- □ **Project3 Requirement**
  - ■ Create ONOS Application (10%)
  - ■ Learning Bridge Function (60%)
  - ■ Flow Rule Regulation (20%)
  - ■ **Submission Naming Convention (10%)**
  - ■ Restrictions

About Submission

- ❑ Files
  - ■ You need to submit all files under the ***bridge-app*** project directory.
  - ■ Zip the whole ***bridge-app*** folder into a .zip file.
    - ● Named: **project3_<studentID>.zip**
- ❑ Submit
  - ■ Upload ".zip" file to New e3
    - ● Named: **project3_<studentID>.zip**
  - ■ Your project will not be scored if you type wrong file name or wrong format.



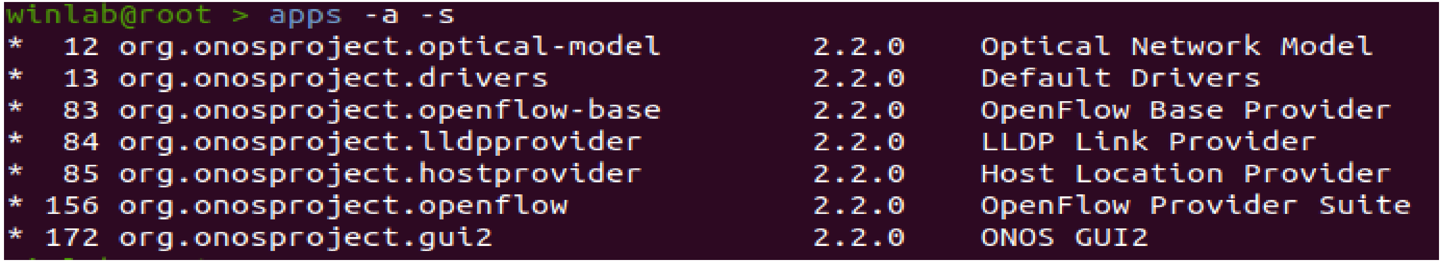
```
sdnfv@sdnfv-VirtualBox:~/bridge-app$ tree
.
├── pom.xml
└── src
    ├── main
    │   └── java
    │       └── nctu
    │           └── winlab
    │               └── bridge
    │                   ├── AppComponent.java
    │                   └── SomeInterface.java
    └── test
        └── java
            └── nctu
                └── winlab
                    └── bridge
                        └── AppComponentTest.java

11 directories, 4 files
```

Outline

- ❑ Overview
- ❑ Build ONOS Application Project
  - ■ Environment Setup
  - ■ Create and Write ONOS Application
  - ■ Compile, Install, and Activate ONOS Application
  - ■ Reinstall ONOS Application
- ❑ Learning Bridge Function
  - ■ Introduction
  - ■ Workflow
- ❑ **Project3 Requirement**
  - ■ Create ONOS Application (10%)
  - ■ Learning Bridge Function (60%)
  - ■ Flow Rule Regulation (20%)
  - ■ Submission Naming Convention (10%)
  - ■ **Restrictions**

Restrictions

- ❑ **ONOS Applications activation**
  - ■ You are only allowed to activate your ***bridge-app*** and the following ONOS applications:



```
winlab@root > apps -a -s
*   12 org.onosproject.optical-model        2.2.0    Optical Network Model
*   13 org.onosproject.drivers              2.2.0    Default Drivers
*   83 org.onosproject.openflow-base        2.2.0    OpenFlow Base Provider
*   84 org.onosproject.lldpprovider         2.2.0    LLDP Link Provider
*   85 org.onosproject.hostprovider         2.2.0    Host Location Provider
*  156 org.onosproject.openflow             2.2.0    OpenFlow Provider Suite
*  172 org.onosproject.gui2                 2.2.0    ONOS GUI2
```

- ❑ You are only allowed to use Java API FlowObjective or FlowRule to install flow rules on the network devices.

Hints

- Use Java API FlowObjective or FlowRule to send Flow-mod
  - You can trace *ReactiveForwarding.java* to find out how can we use Java API to install flow rules
- Make sure to Packet-out when you send Flow-mod
  - Since flow modification message only install flow rule on the switch
- Make sure to cancel request for Packet-in when you deactivate your app
- How to debug:
  (1) Use Logger (Java API) to print out some information on your terminal
  (2) Use Wireshark to capture your packet

References

- ONOS Reactive Forwarding application
  - https://github.com/opennetworkinglab/onos/blob/master/apps/fwd/src/main/java/org/onosproject/fwd/ReactiveForwarding.java
- ONOS Java API
  - http://api.onosproject.org/2.2.0/apidocs/