前言

本来我觉得没必要写这篇教程的，因为关于安装floodlight，mininet构建数据中心拓扑的教程网上就有，何必再写这篇，这是因为我在复现的过程中，发现一些floodlight更新过程中带来的问题，一言以蔽之，就是floodlight在更新，而教程太老了。这篇教程其实就是我实现实验过程中遇到问题和解决问题的总结。

背景

Mininet 采用轻量级的虚拟化技术， 基于 Mininet 研究者能够在笔记本电脑上搭建自定义拓扑的 SDN 网络，并对 SDN 相关的创新设计进行测试和验证。一旦验证成功，就能在实际环境中进行部署。基于 Mininet 构建的仿真测试能够评估多数据中心网络应用设计的效果，为真实数据中心网络应用开发提供参考。

安装前准备工作

（1）Linux：Ubuntu14.04版本 这里提供镜像，各种版本都有
http://pan.baidu.com/s/1skKN5CT
（2）安装JDK，Ant 提供命令行
sudo apt-get install build-essential default-jdk ant python-dev
（3）安装git 提供命令行
sudo apt-get install git

安装Floodlight

下载源

（1）从Github下载并编译Floodlight，提供命令行
git clone git://github.com/floodlight/floodlight.git
（2）git下载速度慢的，可以选择从github上直接下载，然后解压，或者也可以从网盘下载，然后拷进虚拟机即可
http://pan.baidu.com/s/1qYe2s8k

编译安装

cd floodlight
ant
直接编译会出现如下错误，看到如下报错信息

```
lyx@ubuntu:~/floodlight$ ant
Buildfile: /home/lyx/floodlight/build.xml
  [taskdef] Could not load definitions from resource tasks.properties. It could
not be found.

init:
    [mkdir] Created dir: /home/lyx/floodlight/target/bin
    [mkdir] Created dir: /home/lyx/floodlight/target/bin-test
    [mkdir] Created dir: /home/lyx/floodlight/target/lib
    [mkdir] Created dir: /home/lyx/floodlight/target/test

compile:
    [javac] Compiling 538 source files to /home/lyx/floodlight/target/bin
    [javac] javac: invalid target release: 1.8
    [javac] Usage: javac <options> <source files>
    [javac] use -help for a list of possible options

BUILD FAILED
/home/lyx/floodlight/build.xml:145: Compile failed; see the compiler error outpu
t for details.

Total time: 1 second
```

```
lyx@ubuntu:~/floodlight$ ant
Buildfile: /home/lyx/floodlight/build.xml
  [taskdef] Could not load definitions from resource tasks.properties. It could
not be found.

init:
    [mkdir] Created dir: /home/lyx/floodlight/target/bin
    [mkdir] Created dir: /home/lyx/floodlight/target/bin-test
    [mkdir] Created dir: /home/lyx/floodlight/target/lib
    [mkdir] Created dir: /home/lyx/floodlight/target/test

compile:
    [javac] Compiling 538 source files to /home/lyx/floodlight/target/bin
    [javac] javac: invalid target release: 1.8
    [javac] Usage: javac <options> <source files>
    [javac] use -help for a list of possible options

BUILD FAILED
/home/lyx/floodlight/build.xml:145: Compile failed; see the compiler error outpu
t for details.

Total time: 1 second
```

我们发现报错信息题型jdk版本需要1.8，我们看下我们的jdk版本

```
lyx@ubuntu:~/floodlight$ java -version
java version "1.7.0_121"
OpenJDK Runtime Environment (IcedTea 2.6.8) (7u121-2.6.8-1ubuntu0.14.04.3)
OpenJDK 64-Bit Server VM (build 24.121-b00, mixed mode)
```

发现是1.7的，这是因为之前的安装jdk的命令，安装的是default-jdk，而ubuntu14.04的默认jdk是1.7，为什么会出这个问题？

因为本文参照的教程在成稿时，floodlight不是现在1.2的版本，jdk的需求在1.7以下，所以没有问题，所以我们现在需要安装jdk1.8

（1）源码包准备：

首先到官网下载jdk，http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html，下载jdk-8u121-linux-x64.tar.gz

Java SE Development Kit 8u121

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.
Thank you for accepting the Oracle Binary Code License Agreement for Java SE; you may now download this software.

Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.86 MB	jdk-8u121-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.83 MB	jdk-8u121-linux-arm64-vfp-hflt.tar.gz
Linux x86	162.41 MB	jdk-8u121-linux-i586.rpm
Linux x86	177.13 MB	jdk-8u121-linux-i586.tar.gz
Linux x64	159.96 MB	jdk-8u121-linux-x64.rpm
Linux x64	174.76 MB	jdk-8u121-linux-x64.tar.gz
Mac OS X	223.21 MB	jdk-8u121-macosx-x64.dmg
Solaris SPARC 64-bit	139.64 MB	jdk-8u121-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	99.07 MB	jdk-8u121-solaris-sparcv9.tar.gz
Solaris x64	140.42 MB	jdk-8u121-solaris-x64.tar.Z
Solaris x64	96.9 MB	jdk-8u121-solaris-x64.tar.gz
Windows x86	189.36 MB	jdk-8u121-windows-i586.exe
Windows x64	195.51 MB	jdk-8u121-windows-x64.exe

（2）解压源码包

通过终端在/usr/local目录下新建java文件夹，命令行：

sudo mkdir /usr/local/java

然后将下载到压缩包拷贝到java文件夹中，命令行：

进入jdk源码包所在目录

sudo cp jdk-8u121-linux-x64.tar.gz /usr/local/java

然后进入java目录，命令行：

cd /usr/local/java

解压压缩包，命令行：

sudo tar xvf jdk-8u121-linux-x64.tar.gz

然后可以把压缩包删除，命令行：

sudo rm jdk-8u121-linux-x64.tar.gz

（3）设置jdk环境变量

这里采用全局设置方法，它是是所有用户的共用的环境变量

$sudo gedit ~/.bashrc

如下图所示：

File Edit View Search Tools Documents Help

Open Save Undo

.bashrc

```
# Add an "alert" alias for long running commands.  Use like so:
#   sleep 10; alert
alias alert='notify-send --urgency=low -i "$([ $? = 0 ] && echo terminal
echo error)" "$(history|tail -n1|sed -e '\''s/^\s*[0-9]\+\s*//;s/[;&|]\s*
$//'\'')"'

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi
```

打开之后在末尾添加

export JAVA_HOME=/usr/local/java/jdk1.8.0_121

export JRE_HOME=$JAVA_HOME/jreexportCLASSPATH=.:$JAVA_HOME/lib:$JRE_HOME/libexportPATH=$JAVA_HOME/bin:$PATH

重启后我们可以发现jdk已经更新到1.8版本了

```
lyx@ubuntu:~$ java -version
java version "1.8.0_121"
Java(TM) SE Runtime Environment (build 1.8.0_121-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.121-b13, mixed mode)
lyx@ubuntu:~$
```

然后编译

```
lyx@ubuntu:~$ cd floodlight/
lyx@ubuntu:~/floodlight$ ant
```

编译成功

```
compile:
    [javac] Compiling 538 source files to /home/lyx/floodlight/target/bin
    [javac] Note: Some input files use or override a deprecated API.
    [javac] Note: Recompile with -Xlint:deprecation for details.
    [javac] Note: Some input files use unchecked or unsafe operations.
    [javac] Note: Recompile with -Xlint:unchecked for details.
     [copy] Copying 8 files to /home/lyx/floodlight/target/bin
     [copy] Copied 4 empty directories to 1 empty directory under /home/lyx/floo
dlight/target/bin

compile-test:
    [javac] Compiling 87 source files to /home/lyx/floodlight/target/bin-test

dist:
     [echo] Setting Floodlight version: 1.2-SNAPSHOT
     [echo] Setting Floodlight name: floodlight
      [jar] Building jar: /home/lyx/floodlight/target/floodlight.jar
      [jar] Building jar: /home/lyx/floodlight/target/floodlight-test.jar

BUILD SUCCESSFUL
Total time: 58 seconds
```

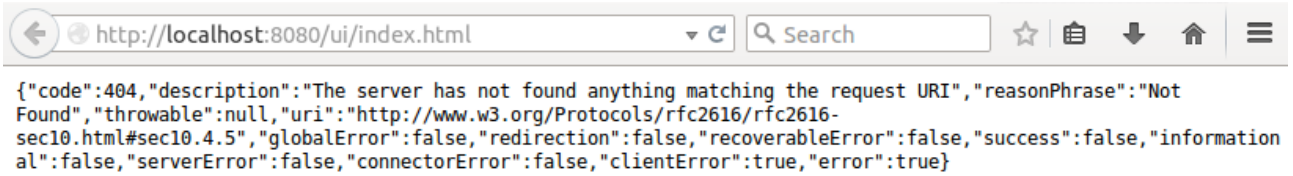
然后启动floodlight，通过命令行

java -jar target/floodlight.jar

启动后，通过浏览器访问floodlight的管理界面，

http://localhost:8080/ui/index.html

结果出现下面的问题



通过查资料发现，似乎是最新的版本的问题，有两个方法解决（但其实目前只能算一种），一种需要通过git Submodule命令进行一系列的设置，通过命令行（这边稍微解释下git Submodule命令，git Submodule 是一个很好的多项目使用共同类库的工具，他允许类库项目做为repository,子项目做为一个单独的git项目存在父项目中，子项目可以有自己的独立的commit，push，pull。而父项目以Submodule的形式包含子项目，父项目可以指定子项目header，父项目中会的提交信息包含Submodule的信息，再clone父项目的时候可以把Submodule初始化。）

但是这种方法虽然可以访问管理界面，但是并不能和mininet进行交互，具体原因不明，所以目前为止还是乖乖用旧版本

git pull origin master

git submodule init

git submodule update

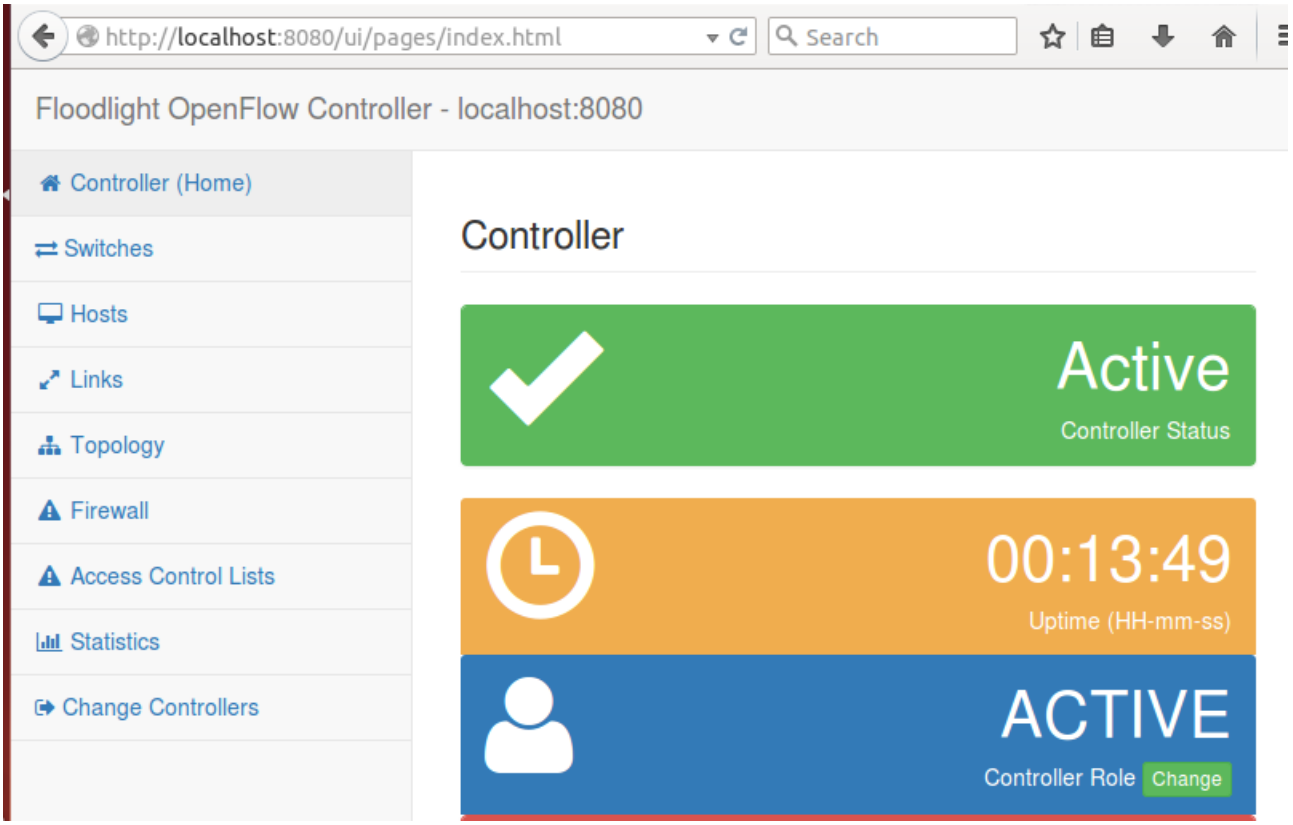
ant

```
^Clyx@ubuntu:~/floodlight$ git pull origin master
From git://github.com/floodlight/floodlight
 * branch            master     -> FETCH_HEAD
Already up-to-date.
lyx@ubuntu:~/floodlight$ git submodule init
Submodule 'src/main/resources/web' (https://github.com/floodlight/floodlight-web
ui) registered for path 'src/main/resources/web'
lyx@ubuntu:~/floodlight$ git submodule update
Cloning into 'src/main/resources/web'...
remote: Counting objects: 1314, done.
remote: Total 1314 (delta 0), reused 0 (delta 0), pack-reused 1314
Receiving objects: 100% (1314/1314), 3.70 MiB | 114.00 KiB/s, done.
Resolving deltas: 100% (353/353), done.
Checking connectivity... done.
Submodule path 'src/main/resources/web': checked out '580bf06fd86bb7ff270019447f
023f9d98e431d9'
lyx@ubuntu:~/floodlight$  ant
```

编译完后启动floodlight

java -jar target/floodlight.jar

再通过http://localhost:8080/ui/index.html访问floodlight的管理界面，发现可以顺利访问



另一种，就是目前为止有效的就是下载0.91版本，即旧版本没有问题，下载地址为

http://www.projectfloodlight.org/download/

the Installation Guide. The source code repository is available on GitHub.

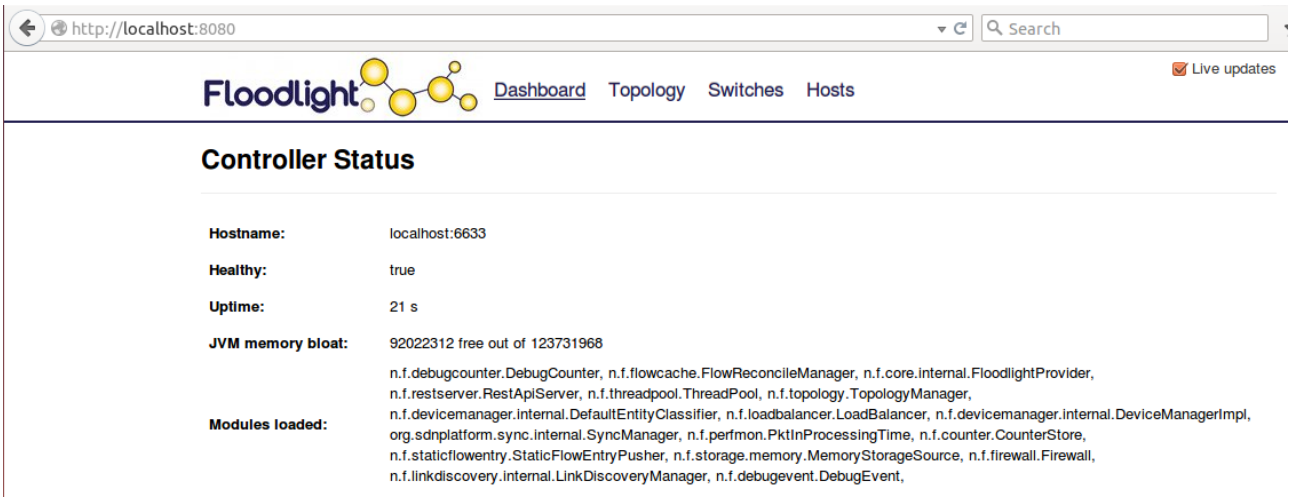
	Source	Release Notes
Nightly	.zip	N/A
Version 1.2	.gz, .zip	1.2 Release Notes
Version 1.1	.gz, .zip	1.1 Release Notes
Version 1.0	.gz, .zip	1.0 Release Notes
Version 0.91	.gz, .zip	0.91 Release Notes
Version 0.90	.gz, .zip	0.90 Release Notes
Version 0.85	.gz, .zip	0.85 Release Notes

然后解压编译

ant

java -jar target/floodlight.jar

再通过http://localhost:8080/ui/index.html访问floodlight的管理界面，发现可以顺利访问



安装mininet

安装完控制器后，我们就需要安装mininet构建拓扑结构了，命令行如下

sudo apt-get install mininet

接着需要构建网络拓扑的脚本fattree.py，

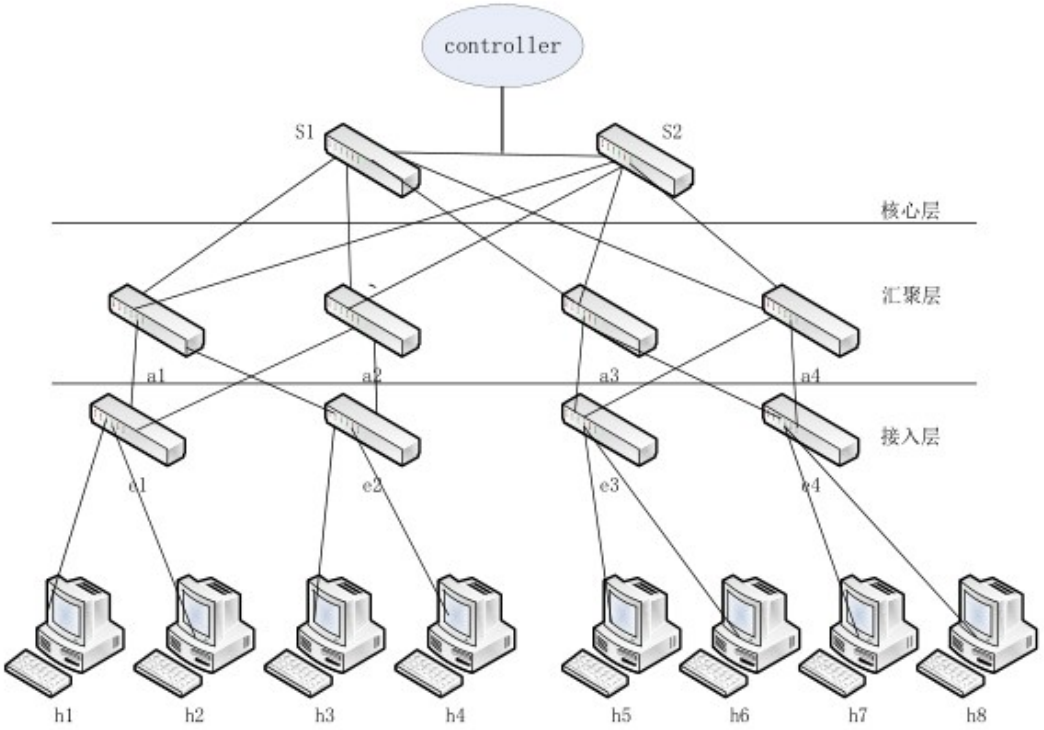


图 3　基于两个数据中心的网络拓扑

资源如下

http://pan.baidu.com/s/1c20519Q

完成后只需要用mininet执行脚本即可

sudo mn –custom /home/lyx/fattree.py –topo mytopo –controller=remote,ip=218.193.113.249,port=6633 –switch ovsk,protocols=OpenFlow10

请根据实际情况将ip为floodlight所在服务器的ip，添加protocols参数指定OpenFlow协议版本。

mn为mininet启动命令。

–mac指定虚拟主机的mac地址顺序编号，若不带此参数则随机编号

–controller指定of交换机的控制器

–switch指定虚拟交换机的类型，ovsk表示虚拟交换机为ovs Kernel mode

–custom指定自定义拓扑文件

–topo指定加载拓扑的名字

执行过程如下图所示：

```
lyx@ubuntu:~/mininet$ sudo mn --custom /home/lyx/mininet/fattree.py --topo mytop
o --controller=remote,ip=192.168.186.137,port=6633 --switch ovsk,protocols=OpenF
low10
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8
*** Adding switches:
S1 S2 a3 a4 a5 a6 e7 e8 e9 e10
*** Adding links:
(S1, a3) (S1, a4) (S1, a5) (S1, a6) (S2, a3) (S2, a4) (S2, a5) (S2, a6) (a3, e7)
 (a3, e8) (a4, e7) (a4, e8) (a5, e9) (a5, e10) (a6, e9) (a6, e10) (e7, h1) (e7,
h2) (e8, h3) (e8, h4) (e9, h5) (e9, h6) (e10, h7) (e10, h8)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8
*** Starting controller
*** Starting 10 switches
S1 S2 a3 a4 a5 a6 e7 e8 e9 e10
*** Starting CLI:
```

```
lyx@ubuntu: ~
lyx@ubuntu:~$ sudo mn --custom /home/lyx/fattree.py --topo mytopo --controller=r
emote,ip=218.193.113.249,port=6633 --switch ovsk,protocols=OpenFlow10
[sudo] password for lyx:
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8
*** Adding switches:
S1 S2 a3 a4 a5 a6 e7 e8 e9 e10
*** Adding links:
(S1, a3) (S1, a4) (S1, a5) (S1, a6) (S2, a3) (S2, a4) (S2, a5) (S2, a6) (a3, e7)
 (a3, e8) (a4, e7) (a4, e8) (a5, e9) (a5, e10) (a6, e9) (a6, e10) (e7, h1) (e7,
h2) (e8, h3) (e8, h4) (e9, h5) (e9, h6) (e10, h7) (e10, h8)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8
*** Starting controller
*** Starting 10 switches
S1 S2 a3 a4 a5 a6 e7 e8 e9 e10
*** Starting CLI:
mininet>
```

Floodlight Dashboard Topology Switches Hosts

Live updates

Disconnected
Ethernet network

Switches (10)

DPID	IP Address	Vendor	Packets	Bytes	Flows	Connected Since
00:00:00:00:00:00:00:01	/218.193.113.249:54095	Nicira, Inc.	0	0	0	4/6/2017, 5:15:57 AM
00:00:00:00:00:00:00:02	/218.193.113.249:54096	Nicira, Inc.	0	0	0	4/6/2017, 5:15:57 AM
00:00:00:00:00:00:00:03	/218.193.113.249:54097	Nicira, Inc.	0	0	0	4/6/2017, 5:15:58 AM
00:00:00:00:00:00:00:04	/218.193.113.249:54101	Nicira, Inc.	0	0	0	4/6/2017, 5:16:00 AM
00:00:00:00:00:00:00:05	/218.193.113.249:54099	Nicira, Inc.	0	0	0	4/6/2017, 5:15:59 AM
00:00:00:00:00:00:00:07	/218.193.113.249:54102	Nicira, Inc.	0	0	0	4/6/2017, 5:16:00 AM
00:00:00:00:00:00:00:06	/218.193.113.249:54105	Nicira, Inc.	0	0	0	4/6/2017, 5:16:04 AM
00:00:00:00:00:00:00:08	/218.193.113.249:54103	Nicira, Inc.	0	0	0	4/6/2017, 5:16:03 AM
00:00:00:00:00:00:00:09	/218.193.113.249:54108	Nicira, Inc.	0	0	0	4/6/2017, 5:16:03 AM
00:00:00:00:00:00:00:0a	/218.193.113.249:54107	Nicira, Inc.	0	0	0	4/6/2017, 5:16:01 AM

要求：

在 Mininet 命令行操作界面通过 iperf 命令对 h1 和 h2，h1 和 h3，h1 和 h5 主机间带宽性能进行分析（也可通过编写测试脚本实现全自动化测试），包括发送速率与接收速率。

实验步骤：

（1）同一交换机内部的主机间连通性及通信带宽测试，在h1和h2之间进行ping操作。

```
mininet> iperf h1 h2
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['6.97 Gbits/sec', '6.98 Gbits/sec']
```

（2）相同汇聚交换机下不同机架的主机间测试，在h1和h3之间进行ping操作。

```
mininet> iperf h1 h3
*** Iperf: testing TCP bandwidth between h1 and h3
*** Results: ['1.58 Gbits/sec', '1.59 Gbits/sec']
```

（3）相同核心交换机不同汇聚交换机下的主机间测试，在h1和h5之间进行ping操作。

```
mininet> iperf h1 h5
*** Iperf: testing TCP bandwidth between h1 and h5
*** Results: ['4.21 Gbits/sec', '4.22 Gbits/sec']
```

参考材料

http://www.sdnlab.com/2909.html

http://www.linuxidc.com/Linux/2015-01/112030.htm

https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/8650780/Floodlight+VM

https://segmentfault.com/a/1190000003076028