# Mininet简介及安装

## Mininet介绍

Mininet是用于在笔记本或其他PC上创建虚拟网络环境的虚拟胡网络仿真工具，只需简单的命令即可快速的在一台机器（虚拟机，云服务器或本机）创建一个具有主机、交换机、控制器和链路的虚拟网络。通过Mininet的CLI（命令行接口）或API能够便利的与虚拟网络进行交互。Mininet支持OpenFlow和OVS，能够进行SDN相关网络的测试开发。

## Mininet主要特性

Mininet作为一个轻量级网络研发和测试平台，主要包含以下特性：

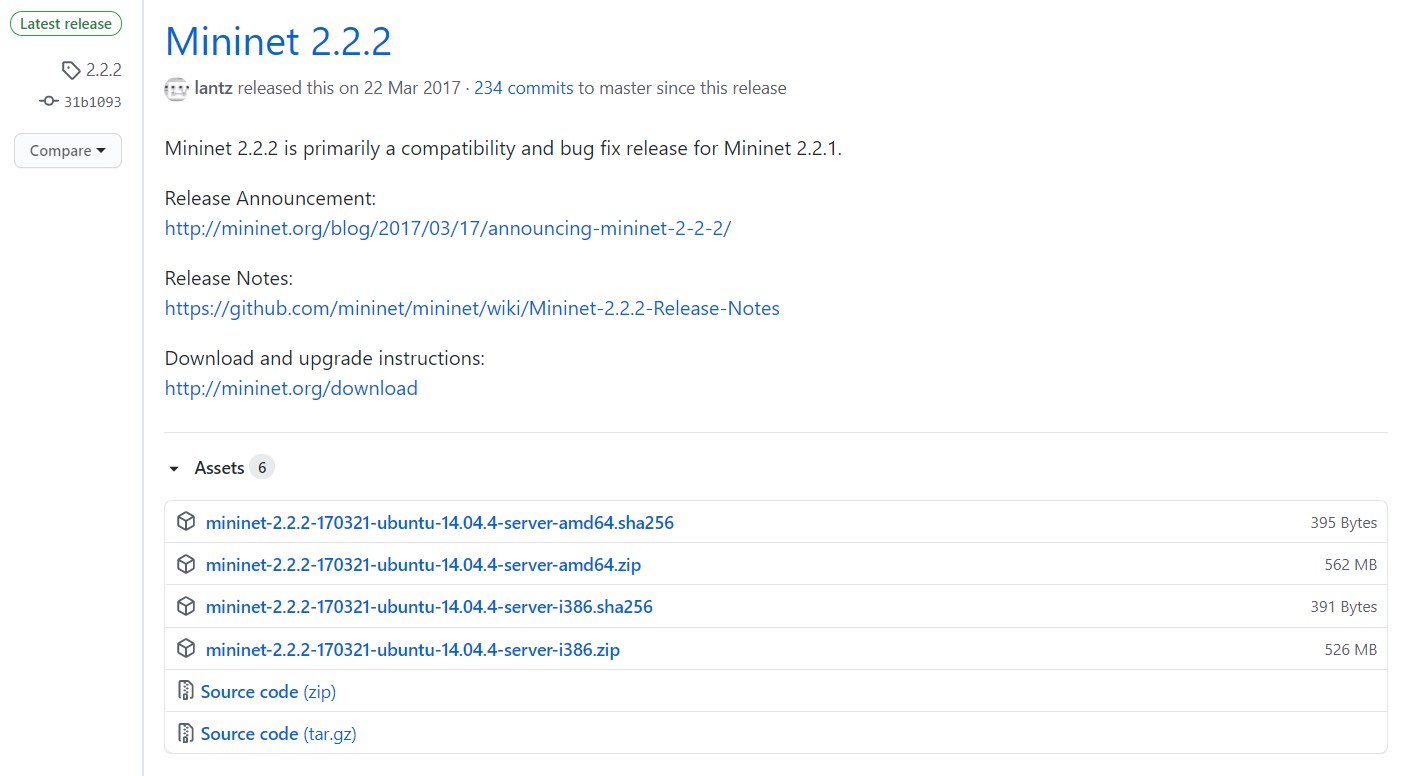
* 提供Python API。
* 支持复杂拓扑和自定义拓扑。
* 支持OpenFlow和OVS等SDN部件。

## Mininet安装

本节介绍Mininet的两种安装方式：使用官方提供的虚拟机镜像以及源码安装。

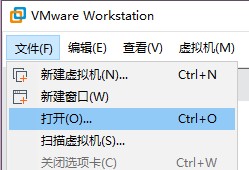
1. **官方虚拟机镜像安装。**
2. 下载虚拟机镜像。<https://github.com/mininet/mininet/releases>

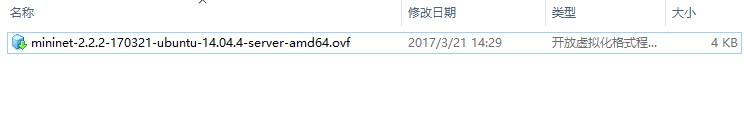
选择所需版本进行下载（本文基于mininet-2.2.2-170321-ubuntu-14.04.4-server-amd64.zip，下图中“Assets”中为下载资源，“.sha56”后缀文件为对应文件的sha256值文件，“amd64.zip”表示Ubuntu虚拟机为64位，"i386.zip"表示Ubuntu需笔记为32位，最后两项位源码文件）。

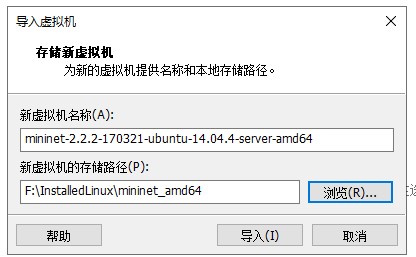


1. 安装一个虚拟化软件，本文档基于VMware Workstation。
2. 打开VMware Workstation，依次选择：
3. “文件”按钮。
4. “打开”选项。
5. 选择解压目录中的“.ovf”文件。
6. “导入”按钮。

即可安装完成，如下图。







说明：安装完成后开启虚拟机即可，默认用户名为“mininet”，密码为“mininet”。该操作系统root用户密码使用“sudo passwd root”进行设置。

1. **源码安装。**
2. 更新软件

# apt-get update

# apt-get upgrade

1. 安装git

# apt install git

1. 从GitHub获取源码

# git clone git://github.com/mininet/mininet

1. 进入mininet目录

# cd mininet

1. 查看现有版本

# git tag

1. 选择所需版本（本文使用2.2.2）

# git checkout -b 2.2.2 2.2.2

1. 执行安装程序

# ./util/install.sh -a

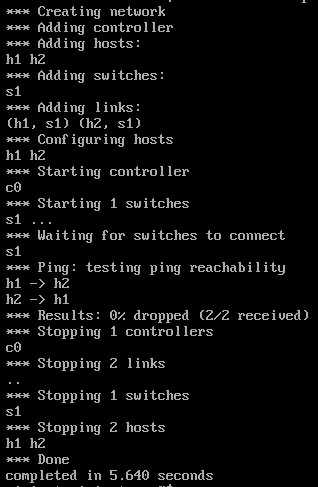
安装选项说明：

* “-a”：安装官方VM虚拟机提供的所有工具，包括Open vSwitch以及OpenFlow wireshark和POX。默认情况下这些工具会安装在home目录。
* “-nfv”：安装Mininet、OpenFlow reference交换机和Open vSwitch。
* “-s mydir”：指定安装目录。

1. 安装完成后可通过简单的命令测试Mininet基本功能。

# sudo mn –-test pingall

输出如下图所示：

****

1. **目录结构介绍**

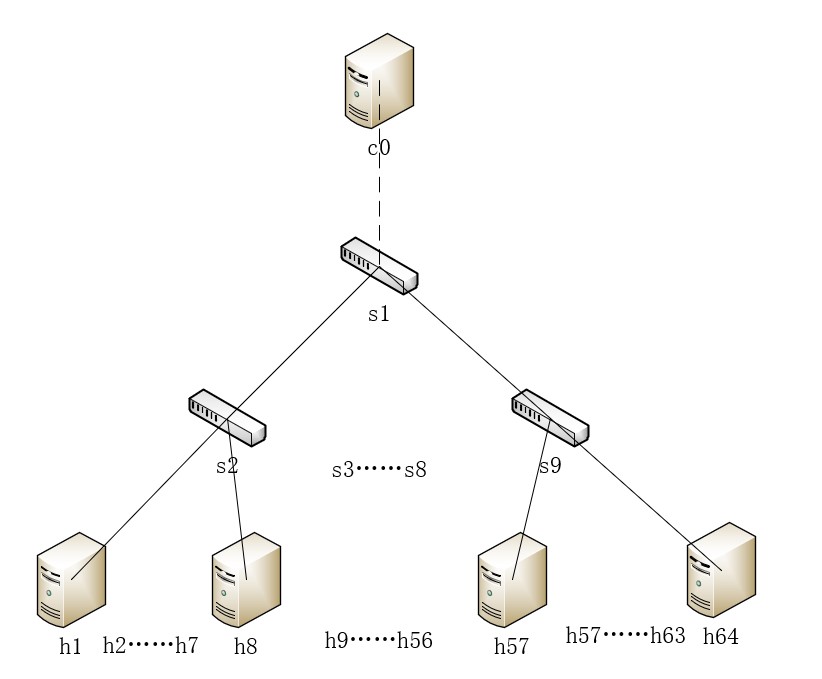
* **mininet/bin/：目录中只有一个“mn”文件，是mininet执行文件。**
* **mininet/util/：内有多个子目录以及文件，主要存储一些工具脚本。**
* **mininet/custom/：目录中有两个文件：READEME和一个后缀为“.py”的脚本文件，脚本文件定义了一个网络拓扑（示例），本目录用于存储自定义网络的Python文件。**
* **mininet/examples/：该目录存放官方提供的Mininet示例脚本。**

## Mininet基本用途

1. 创建网络

该节通过一个简单Mininet命令创建一个简单的网络并对使用到的CLI参数进行说明。

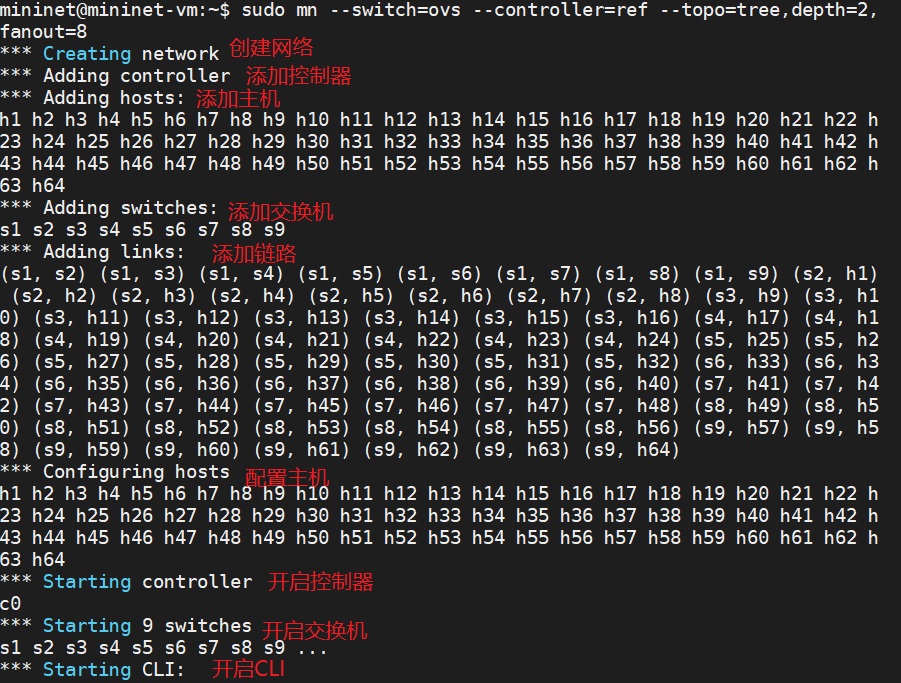
1. 通过命令“sudo mn –switch ovs –controller ref –topo tree,depth=2,fanout=8 –test pingall”即可创建一个简单的网络环境，该网络拓扑是树形结构，包含一个控制器，九个交换机和64台主机，网络拓扑图如下：



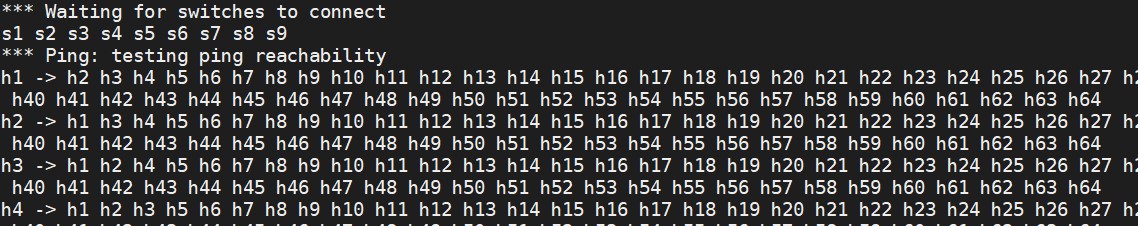
命令行参数说明：

* “--switch ovs”：指定使用的交换机为OVS交换机。
* “--controller ref”：控制器类型为ref（OpefFlow/Standford reference controller）。
* “--topo tree,depth=2,fanout=8”：拓扑结构为树形，深度为2，每个节点有八个子节点。
* “--test pingall”：网络建立后，在每台主机上ping其他主机测试连通性。

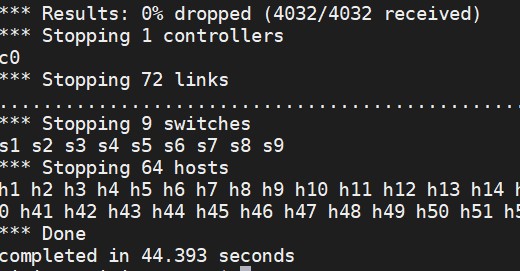
测试结果如下：



**创建网络过程图**

****

**pingall信息图**

****

**ping结果图**

1. 与网络交互

使用命令“sudo mn –switch ovs –controller ref –topo tree,depth=2,fanout=8”会创建上述网络并进入Mininet CLI界面，在该界面可以使用Mininet命令与网络中的机器进行交互。

* mininet> h2 ping h3 ：使用网络或中主机h2 ping h3，验证两个主机之间连通性。
* “mininet> h2 python -m SimpleHTTPServer 80 >& /tmp/http.log &” ： 在主机h2上创建监听80端口的HTTP服务器。
* “mininet> h3 wget -O – h2”：在主机h3访问h2中的HTTP服务器。

1. 自定义网络

使用Mininet的Python API自定义网络环境。

例如：

from mininet.net import Mininet

from mininet.topolib import TreeTopo

tree = TreeTopo(depth=2,fanout=2) #创建拓扑

net = Mininet(topo=tree4) #创建网络

net.start() #开启网络

h1, h4 = net.hosts[0], net.hosts[3] #获取主机

print h1.cmd(‘ping -c1 %s’ % h4.IP()) #在主机h1执行“ping -c1 10.0.0.2”

net.stop() #关闭网络

## CLI高级启动选项

1. 运行测试

通过“--test”选项，不需要每次都进入CLI执行操作,而是会自动执行选项对应的操作并将结果输出。

* “sudo mn --test pingpair”：创建一个mininal拓扑的网络，使用OpenFlow reference控制器，并且执行ping操作，最后关闭网络和控制器。
* “sudo mn --test iperf“：创建一个mininal拓扑的网络，在其中一台主机运行一个iperf服务器，在另一台主机运行iperf客户端，并解析网络带宽。

1. 修改拓扑大小和类型

默认拓扑是一个交换机连接两台主机。可以使用--topo以及创建拓扑的参数。例如：

* “sudo mn --test pingall --topo single,3”：创建一个拓扑有一个交换机连接到三台主机上，并执行pingall，返回执行结果并关闭网络。
* “sudo mn --test pingall --topo linear,4“：创建一个线性拓扑，四台交换机都连接到一台主机，并且四台交换机连接到一条线上。

1. 链路改变

Mininet 2.0之后的版本可以设置链路参数。

$ sudo mn –link tc,bw=10,delay=10ms //在linux命令行执行

mininet>ipref

…

mininet>h1 ping -c10 h2

每条链路的delay是10ms，那么RTT会是40ms，因为ICMP请求穿过两条链路（一条到交换机，一条到目标）并且ICMP响应也经过两条链路。

1. 信息输出级别

默认的输出信息等级是“info”，该等级会打印Mininet启动和停止时做了什么。完整的“debug”输出可以使用“-v“参数指定。例如：

$sudo mn -v debug

…

mininet>exit

除此之外还有“ouput“和”warning“选项。

1. 自定义网络拓扑

可以通过Python API实现方便的自定义网络拓扑，在mininet安装目录中custom/topo-2sw-2host.py中是一个示例，它包含两台相连的交换机以及每一台交换机连接一台主机。当提供了自定义的mininet文件，它可以增添新的拓扑逻辑、交换机类型并且可以在命令行进行测试，例如： “sudo mn --custom ~/mininet/custom/topo-2sw-2host.py --topo mytopo --test pingall“。

1. 设置ID = MAC

默认情况下，主机启动时会随机分配一个MAC地址。这对调试很不友好。此时--mac选项就很有用，它可以将主机的MAC和IP地址设置为更短、唯一且易读的ID。

注：交换机由Linux指定的MAC数据口地址依然是随机的。因为可以通过OpenFlow对交换机的MAC进行指定。

1. Xterm显示

为了复杂的测试，可以在启动mininet时使用参数-x打开多个窗口，每个窗口对应一台主机或交换机。在对应的窗口执行命令即可。例如：“sudo mn -x“会打开四个窗口，分别对应控制器，交换机和两台主机。也可以在mininet CLI中使用“名字 xterm &”的方式打开一个xterm窗口。例如”h1 xterm &“或”xterm h1“会为主机h1打开一个xterm窗口（可为一个机器打开多个xterm窗口）。

1. 其他交换机类型

使用“--switch“选项可以指定交换机类型。例如：

* “sudo mn --switch user --test iperf”：指定交换机为用户态交换机。 注：用户态交换机使用iperf报告的带宽会比前面的内核交换机更低，如果执行前面的ping操作会看到更高的延迟，因为现在需要额外将数据包从内核传到用户态。
* “sudo mn --switch ovsk --test iperf”：这是另一种交换机--Open vSwitch（OVS），他已被提前安装到Mininet VM中，他的iperf TCP带宽应该与OpenFlow内核交换机相似甚至更快。

1. 设置NameSpace

默认情况下，只有主机位于各自的NameSpace，而交换机和控制器都在 root NameSpace下。可以使用“--innamespace”将交换机放到自己的NameSpace。例：“sudo mn --innamespace --switch user“。与之前使用loopback与控制器进行通信不同，现在交换机会通过单独的桥接控制连接与控制器通信。孤立的讲，这个选项没什么用处，但是确实为如何隔离不同的交换机提供了一个例子。

注意：该选项对Open vSwitch不起作用。