LISA使用说明

1. 系统概况

如下图，为整个系统概况图，从下到上主要包含三个部分：

1. Mininet：仿真网络，在Mininet中启动连接N个控制器
2. IR：中间表示层，本系统的核心，用于解除冲突
3. Ryu：N个控制器，每个控制器携带用户意图



整个工程的代码构架：

LISA

----IR

--------util

------------utils.py

--------inst.py

--------lisa\_types.py

--------middle\_end.py

--------networks.py

--------of\_inst.py

----ryu-patch

--------network\_aware.py

--------network\_monitor.py

--------route\_lisa\_1.py

--------route\_lisa\_2.py

----sample\_topo（一些示例拓扑）

--------sample\_topo.py（论文中的四个交换机的拓扑）

--------stanford\_topo.py（斯坦福大学的拓扑）

1. 环境搭建

Ryu：版本没有要求，不过建议4.4.2

直接使用pip install ryu即可安装

Mininet：建议2.2.0

参见http://www.cnblogs.com/cotyb/p/4633329.html

Openvswitch：建议2.4.0  
 参见https://github.com/cotyb/enhancement-ryu/blob/master/ovs\_recompile.sh

1. 如何使用
2. 首先启动指定了Mininet中拓扑的python文件，比如sample\_topo.py，开启命令为：python sample\_topo.py –num=N，特别需要注意的是参数num，指定了控制器的个数（默认是两个），而且这些控制器连接Mininet的端口从6633开始，每一个控制器加1，比如现在需要连接N个控制器。
3. 需要有N个控制器启动，源代码中的route\_lisa\_1是一个master控制器，route\_lisa\_2是一个slave控制器，如果需要增加新的控制器，只需要复制route\_lisa\_2文件，修改文件名为route\_lisa\_3等
4. 指定用户意图：在每个route\_lisa\_n文件的\_packet\_in\_handler()函数中指定information变量为用户意图
5. 启动控制器，因为所有控制器的默认连接端口为6633，所以master控制器的启动可以直接使用ryu-manger --observe-links route\_lisa\_1.py，对于每一个slave控制器来说需要指定连接端口，ryu-manger –observe-links –ofp-tcp-listen-port=6634 route\_lisa\_2.py，对于其他的slave控制器，需要指定另外的端口
6. 在Mininet中指定某个主机ping另外一个主机，比如：h1 ping h2
7. 注意：由于ryu的拓扑发现不是很稳定，所以在每次手动的时候，将ryu-patch文件夹里面的network\_aware.py中192行的数字改成交换机的数目。