SDN Live Demo

1. 基本检查

1.1 输出网络拓扑结构

- 何时输出：没有具体限定，可以在**每次拓扑结构更新时**输出，也可以**在controller收到一个新的包时**输出
- 输出格式：没有具体要求，只要**能清晰表明网络拓扑结构**即可
- 例子：在以下例子中，在controller获得`arp require`时输出当前网络拓扑结构，采用`点:{边}`的表示图形式

```
Got ARP require!
switch_1
Edge: switch_2(82:5c:b3:9a:3d:d4 2/2) <-> switch_1(42:10:6d:35:53:46 1/2)
Edge: switch_1(42:10:6d:35:53:46 1/2) <-> switch_2(82:5c:b3:9a:3d:d4 2/2)
Edge: host_00:00:00:00:00:01(86:4b:74:dc:1b:e8 1/1) <-> switch_1(86:4b:74:dc:1b:
e8 1/1)
switch_3
Edge: switch_3(82:ff:1d:0a:22:c7 3/2) <-> switch_2(de:84:88:21:20:30 2/3)
Edge: switch_2(de:84:88:21:20:30 2/3) <-> switch_3(82:ff:1d:0a:22:c7 3/2)
Edge: host_00:00:00:00:00:03(62:54:40:be:9a:6f 3/1) <-> switch_3(62:54:40:be:9a:
6f 3/1)
switch_2
Edge: switch_2(82:5c:b3:9a:3d:d4 2/2) <-> switch_1(42:10:6d:35:53:46 1/2)
Edge: switch_1(42:10:6d:35:53:46 1/2) <-> switch_2(82:5c:b3:9a:3d:d4 2/2)
Edge: switch_3(82:ff:1d:0a:22:c7 3/2) <-> switch_2(de:84:88:21:20:30 2/3)
Edge: switch_2(de:84:88:21:20:30 2/3) <-> switch_3(82:ff:1d:0a:22:c7 3/2)
Edge: host_00:00:00:00:00:02(4e:30:14:c5:5e:e3 2/1) <-> switch_2(4e:30:14:c5:5e:
e3 2/1)
host_00:00:00:00:00:01
Edge: host_00:00:00:00:00:01(86:4b:74:dc:1b:e8 1/1) <-> switch_1(86:4b:74:dc:1b:
e8 1/1)
host_00:00:00:00:00:02
Edge: host_00:00:00:00:00:02(4e:30:14:c5:5e:e3 2/1) <-> switch_2(4e:30:14:c5:5e:
```

1.2 输出最短路

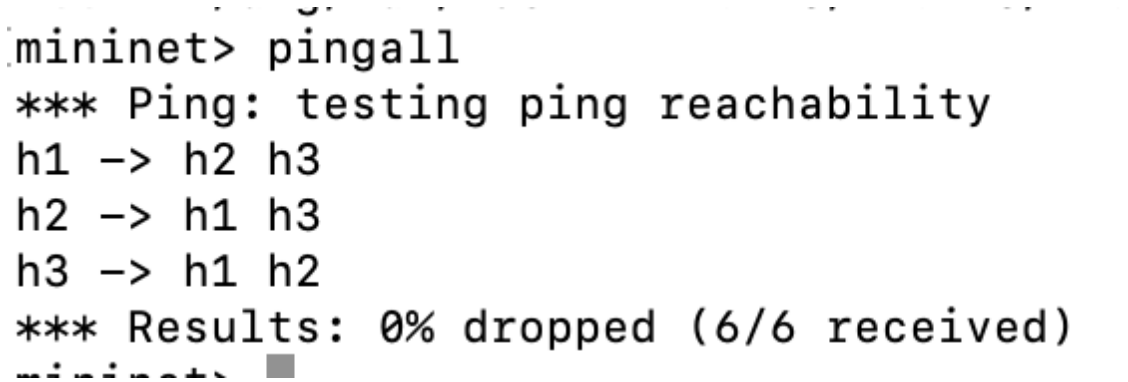
- 何时输出：在mininet中输入`ping`（*所有的`ping`请都指定参数`-c 1`只进行一次`ping`操作*）命令后，controller应该输出从一主机到另一主机的最短路
- 输出格式：没有特殊要求，以下例子中的两种方式皆可（以`linear 3`的拓扑为例）
- 例子1：简易版（只包含路径信息）

```
Node: host_00:00:00:00:00:01    Node: host_00:00:00:00:00:03
distance: 4
host_00:00:00:00:00:01 -> switch_1 -> switch_2 -> switch_3 -> host_00:00:00:00:0
0:03
```

- 例子2：具体（含switch的port信息）

```
Edge1
Edge: host_00:00:00:00:00:01(86:4b:74:dc:1b:e8 1/1) <-> switch_1(86:4b:74:dc:1b:
e8 1/1)
Edge2
Edge: switch_1(42:10:6d:35:53:46 1/2) <-> switch_2(82:5c:b3:9a:3d:d4 2/2)
('switch_1', 1, 2)
Edge1
Edge: switch_1(42:10:6d:35:53:46 1/2) <-> switch_2(82:5c:b3:9a:3d:d4 2/2)
Edge2
Edge: switch_2(de:84:88:21:20:30 2/3) <-> switch_3(82:ff:1d:0a:22:c7 3/2)
('switch_2', 2, 2)
Edge1
Edge: switch_2(de:84:88:21:20:30 2/3) <-> switch_3(82:ff:1d:0a:22:c7 3/2)
Edge2
Edge: host_00:00:00:00:00:03(62:54:40:be:9a:6f 3/1) <-> switch_3(62:54:40:be:9a:
6f 3/1)
('switch_3', 2, 1)
```

1.3 网络连通

- `pingall`：在mininet中输入此命令，应可以看到各个host之间互如下
  - 

  - 若在自测中发现第一次ping丢包，之后不会丢包的情况，可尝试以下解决方案
    - 删除高亮这一行

      ```
      actions = [datapath.ofproto_parser.OFPActionOutput(port)]
      ofctl.set_flow(cookie=0, priority=0,
                     dl_type=ether_types.ETH_TYPE_IP,
                     dl_vlan=VLANID_NONE,
                     dl_dst=dl_dst,
                     actions=actions)
      ```

      from NekoNull1
- `ping` 不存在地址：不应显示ping通

  ```
  mininet> h1 ping 10.0.0.100 -c 1
  PING 10.0.0.100 (10.0.0.100) 56(84) bytes of data.
  From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
  ```

2. 场景改变

只使用`link up/down`命令进行**有环网络拓扑**的改变，在某一条链路断开之后，网络仍能ping通（具体测试及输出可参考**1. 基本检查**）

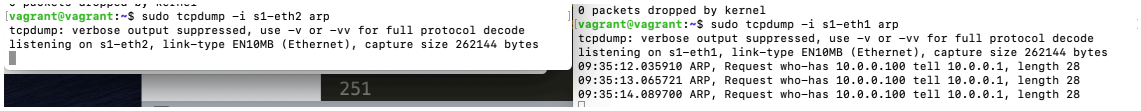
3. 附加功能

在附加功能的展示上，请使用`tcpdump`命令进行辅助说明。控制器**输出生成树**可参考**输出网络拓扑结构**，无特殊要求，只要能清楚展示生成树即可。

3.1 附加功能展示流程

1. 用`tcpdump`监测任意（指定）若干接口的arp包信息
2. `ping`一个不存在的地址产生arp广播（**只ping一次**）
3. 查看网络接口收到包的情况
4. 断掉某一条link后，能重新输出拓扑结构和新的生成树，重复步骤（1-3）

3.2 可能结果（以`triangle 3`为例）

- 广播风暴：监听接口持续收到ARP包
  - ```
    [vagrant@vagrant:~$ sudo tcpdump -i s1-eth2 arp
    tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
    listening on s1-eth2, link-type EN10MB (Ethernet), capture size 262144 bytes
    09:23:18.337752 ARP, Request who-has 10.0.0.100 tell 10.0.0.1, length 28
    09:23:18.358332 ARP, Request who-has 10.0.0.100 tell 10.0.0.1, length 28
    09:23:18.365066 ARP, Request who-has 10.0.0.100 tell 10.0.0.1, length 28
    09:23:18.380911 ARP, Request who-has 10.0.0.100 tell 10.0.0.1, length 28
    09:23:18.387595 ARP, Request who-has 10.0.0.100 tell 10.0.0.1, length 28
    09:23:18.404394 ARP, Request who-has 10.0.0.100 tell 10.0.0.1, length 28
    09:23:18.413947 ARP, Request who-has 10.0.0.100 tell 10.0.0.1, length 28
    09:23:21.013942 ARP, Request who-has 10.0.0.100 tell 10.0.0.1, length 28
    09:23:21.021639 ARP, Request who-has 10.0.0.100 tell 10.0.0.1, length 28
    09:23:21.033578 ARP, Request who-has 10.0.0.100 tell 10.0.0.1, length 28
    09:23:21.055144 ARP, Request who-has 10.0.0.100 tell 10.0.0.1, length 28
    09:23:21.056076 ARP, Request who-has 10.0.0.100 tell 10.0.0.1, length 28
    09:23:21.067292 ARP, Request who-has 10.0.0.100 tell 10.0.0.1, length 28
    ```
- controller直接丢弃广播包（***不算成功实现***）：除了与发出命令的host相连的网络接口可以收到arp request以外，其余收不到arp request
  - 

- 使用生成树避免了广播风暴：所有接口只收到有限个arp请求
    ```

```
mininet> h1 ping h2 -c 1
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.337 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.337/0.337/0.337/0.000 ms
mininet>
```

```
vagrant@vagrant:~$ sudo tcpdump -i s1-eth2 arp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on s1-eth2, link-type EN10MB (Ethernet), capture size 262144 bytes
09:52:09.894404 ARP, Request who-has 10.0.0.2 tell 10.0.0.1, length 28
09:52:09.894501 ARP, Reply 10.0.0.2 is-at 00:00:00:00:00:02 (oui Ethernet), leng
th 28
09:52:14.954296 ARP, Request who-has 10.0.0.1 tell 10.0.0.2, length 28
09:52:14.955543 ARP, Reply 10.0.0.1 is-at 00:00:00:00:00:01 (oui Ethernet), leng
th 28
```

```
vagrant@vagrant:~$ sudo tcpdump -i s1-eth1 arp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on s1-eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
09:52:09.894303 ARP, Request who-has 10.0.0.2 tell 10.0.0.1, length 28
09:52:09.894524 ARP, Reply 10.0.0.2 is-at 00:00:00:00:00:02 (oui Ethernet), leng
th 28
09:52:14.954305 ARP, Request who-has 10.0.0.1 tell 10.0.0.2, length 28
09:52:14.954316 ARP, Reply 10.0.0.1 is-at 00:00:00:00:00:01 (oui Ethernet), leng
th 28
```

-