第六周实验报告

生成树机制实验

2015K8009922021

李一苇

一、实验内容

- 基于已有代码，实现生成树运行机制，对于给定拓扑(four_node_ring.py)，计算输出相应状态下的最小生成树拓扑
- 自己构造一个不少于 6 个节点，链路冗余度不小于 2 的拓扑，节点和端口的命名规则可参考 four_node_ring.py，使用 stp 程序计算输出最小生成树拓扑

二、实验流程

本实验只需实现 stp.c 里的 stp_handle_config_packet 函数

如果 STP 包优先级比端口 config 高：

- ○ 替换端口的 config 信息
- ○ 更新节点的根端口和 cost 信息
- ○ 更新剩余端口的 config 信息
- ○ 停止节点的 hello 计时器
- ○ 每个指定端口转发新的 config 包

如果 STP 包优先级更低：

- ○ 直接从本端口转发自己的 config 信息

需要注意的点：

- 所有涉及 Config 的比较都需要用 ntoh 系列函数进行大小端转化
  优先级函数实现如下：

```
bool is_config_higher(stp_port_t *p, struct stp_config *config) {
    if (ntohll(config->root_id) < p->designated_root) return true;
    else if (ntohll(config->root_id) > p->designated_root) return false;
    else if (ntohl(config->root_path_cost) < p->designated_cost) return true;
    else if (ntohl(config->root_path_cost) > p->designated_cost) return false;
    else if (ntohll(config->switch_id) < p->designated_switch) return true;
    else if (ntohll(config->switch_id) > p->designated_switch) return false;
    else if (ntohs(config->port_id) < p->designated_port) return true;
    else if (ntohs(config->port_id) > p->designated_port) return false;
    else log(INFO, "config equally comparing: impossible");
    return false;
}
```

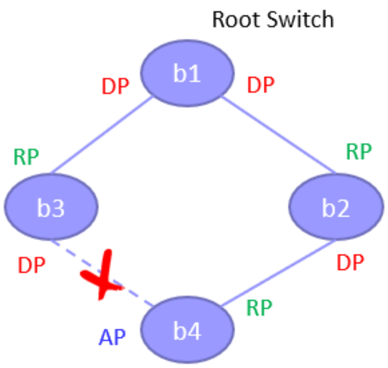
- 如何判定指定端口：在非指定端口中，其网段通过本节点到根节点的开销比通过对端节点的开销小。前者可由 stp->root_path_cost 得到，后者无法得知。但如果条件成立，则一定有 stp->root_path_cost < port_i->designated_cost。因为网段的耗费一定是二者的最小值。
- 何时停掉 hello 计时器：因为初识情况下都是根节点，如果某端口遇到更优的 config 信息，则证明自己一定不是根节点，所以只要 config 优先级高，则停掉 hello 计时器，这一句话一定不会错。
- 不会形成 STP 包的转发风暴：如果收到的 config 优先级低，的确存在回传 config 请求，以便确认对端更新为高优先级；当收到 config 优先级高时，主机从所有 DP 转发 config，而生成树的叶子主机不存在 DP，因此转发中止。

三、实验结果和分析

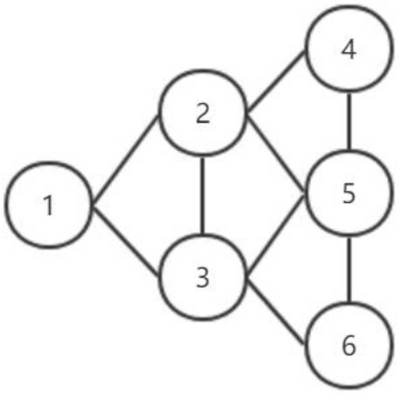
用给定的四结点环路验证 STP 函数如下

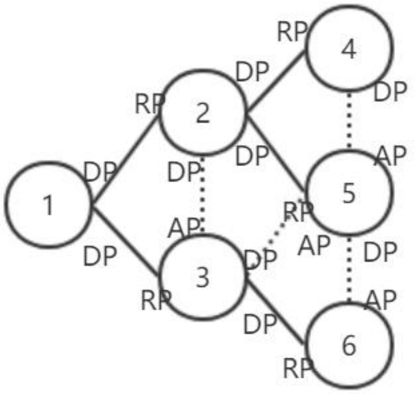
```
NODE b1 dumps:
INFO: this switch is root.
INFO: port id: 01, role: DESIGNATED.
INFO:   designated ->root: 0101, ->switch: 0101, ->port: 01, ->cost: 0.
INFO: port id: 02, role: DESIGNATED.
INFO:   designated ->root: 0101, ->switch: 0101, ->port: 02, ->cost: 0.

NODE b2 dumps:
INFO: non-root switch, designated root: 0101, root path cost: 1.
INFO: port id: 01, role: ROOT.
INFO:   designated ->root: 0101, ->switch: 0101, ->port: 01, ->cost: 0.
INFO: port id: 02, role: DESIGNATED.
INFO:   designated ->root: 0101, ->switch: 0201, ->port: 02, ->cost: 1.

NODE b3 dumps:
INFO: non-root switch, designated root: 0101, root path cost: 1.
INFO: port id: 01, role: ROOT.
INFO:   designated ->root: 0101, ->switch: 0101, ->port: 02, ->cost: 0.
INFO: port id: 02, role: DESIGNATED.
INFO:   designated ->root: 0101, ->switch: 0301, ->port: 02, ->cost: 1.

NODE b4 dumps:
INFO: non-root switch, designated root: 0101, root path cost: 2.
INFO: port id: 01, role: ROOT.
INFO:   designated ->root: 0101, ->switch: 0401, ->port: 02, ->cost: 1.
INFO: port id: 02, role: ALTERNATE.
INFO:   designated ->root: 0101, ->switch: 0301, ->port: 02, ->cost: 1.
```

即得到课件中的网络结构：



自己构造六结点拓扑结构如下左图，显然包含两个以上的冗余度。





STP 程序给出的输出如下，等价于右上的拓扑图

```
NODE b1 dumps:
INFO: this switch is root.
INFO: port id: 01, role: DESIGNATED.
INFO:   designated ->root: 0101, ->switch: 0101, ->port: 01, ->cost: 0.
INFO: port id: 02, role: DESIGNATED.
INFO:   designated ->root: 0101, ->switch: 0101, ->port: 02, ->cost: 0.

NODE b2 dumps:
INFO: non-root switch, designated root: 0101, root path cost: 1.
INFO: port id: 01, role: ROOT.
INFO:   designated ->root: 0101, ->switch: 0101, ->port: 01, ->cost: 0.
INFO: port id: 02, role: DESIGNATED.
INFO:   designated ->root: 0101, ->switch: 0201, ->port: 02, ->cost: 1.
INFO: port id: 03, role: DESIGNATED.
INFO:   designated ->root: 0101, ->switch: 0201, ->port: 03, ->cost: 1.
INFO: port id: 04, role: DESIGNATED.
INFO:   designated ->root: 0101, ->switch: 0201, ->port: 04, ->cost: 1.

NODE b3 dumps:
INFO: non-root switch, designated root: 0101, root path cost: 1.
```

```
INFO: port id: 01, role: ROOT.
INFO:   designated ->root: 0101, ->switch: 0101, ->port: 02, ->cost: 0.
INFO: port id: 02, role: ALTERNATE.
INFO:   designated ->root: 0101, ->switch: 0201, ->port: 02, ->cost: 1.
INFO: port id: 03, role: DESIGNATED.
INFO:   designated ->root: 0101, ->switch: 0301, ->port: 03, ->cost: 1.
INFO: port id: 04, role: DESIGNATED.
INFO:   designated ->root: 0101, ->switch: 0301, ->port: 04, ->cost: 1.

NODE b4 dumps:
INFO: non-root switch, designated root: 0101, root path cost: 2.
INFO: port id: 01, role: ROOT.
INFO:   designated ->root: 0101, ->switch: 0201, ->port: 03, ->cost: 1.
INFO: port id: 02, role: DESIGNATED.
INFO:   designated ->root: 0101, ->switch: 0401, ->port: 02, ->cost: 2.

NODE b5 dumps:
INFO: non-root switch, designated root: 0101, root path cost: 2.
INFO: port id: 01, role: ROOT.
INFO:   designated ->root: 0101, ->switch: 0201, ->port: 04, ->cost: 1.
INFO: port id: 02, role: ALTERNATE.
INFO:   designated ->root: 0101, ->switch: 0301, ->port: 03, ->cost: 1.
INFO: port id: 03, role: ALTERNATE.
INFO:   designated ->root: 0101, ->switch: 0401, ->port: 02, ->cost: 2.
INFO: port id: 04, role: DESIGNATED.
INFO:   designated ->root: 0101, ->switch: 0501, ->port: 04, ->cost: 2.

NODE b6 dumps:
INFO: non-root switch, designated root: 0101, root path cost: 2.
INFO: port id: 01, role: ROOT.
INFO:   designated ->root: 0101, ->switch: 0301, ->port: 04, ->cost: 1.
INFO: port id: 02, role: ALTERNATE.
INFO:   designated ->root: 0101, ->switch: 0501, ->port: 04, ->cost: 2.
```