

网络路由实验

孙佳钰 2015K8009929051

2018 年 12 月 2 日

1 实验内容

OSPF 主要工作为使每个节点都向外通告自己的链路状态信息，从而使网络中每个路由器都有完整的网络拓扑，然后自行计算生成到每个节点的最短路径。本次实验需要实现简化版的 OSPF 协议，内容有：

- 对第一次实验各节点收集到的链路状态数据库进行最短路径的计算，计算出从本节点到各节点的最短路径中下一跳的网关。
- 对超过一段时间没有更新的数据库记录进行老化操作。

2 实验流程

由于代码太多，故报告中没有加入完整代码，完整代码可见附件。

2.1 最短路径计算

1. 采用邻接矩阵的方式储存图，利用 Dijkstra 算法计算到各节点的最短路径。返回各节点到本节点最短路径的下一跳路由 id。

```
1 #define VNUM 4
2 u32 v_list[VNUM];
3 int e_matrix[VNUM][VNUM];
4
5 void Dijkstra(int *path) {
6     u8 dist[VNUM];
7     memset(dist, UINT8_MAX, VNUM);
8     int visited[VNUM] = {0};
9     dist[0] = 0;
10    visited[0] = 1;
```

```

11     for (int i = 0; i < VNUM; i++) {
12         int u = min_dist(dist, visited);
13         visited[u] = 1;
14         for (int j = 0; j < VNUM; j++) {
15             if (0 == visited[j] && e_matrix[u][j] > 0 && \
16                 dist[u] + e_matrix[u][j] < dist[j]) {
17                 dist[j] = dist[u] + e_matrix[u][j];
18                 path[j] = u;
19             }
20         }
21     }
22 }
23
24 void *calculating_rtable_thread(void *param) {
25     while (1) {
26         fprintf(stdout, "DEBUG: calculating rtable.\n");
27         init_graph();
28         .....
29         int path[VNUM] = {0};
30         path[0] = -1;
31         Dijkstra(path);
32         for (int t1 = 1; t1 < VNUM; t1++) {
33             if (path[t1] != 0 && path[path[t1]] != 0) {
34                 path[t1] = path[path[t1]];
35                 t1--;
36             }
37         }
38         .....
39     }

```

2. 给路由条目增加一个有效域，每次更新时将网关为 0，即从内核中读入的路由条目均置为有效；其余节点均置为无效。从路径长度最短，即相邻节点开始遍历，依次查看每个节点的 lsa 数组信息，若路由表中没有对应条目，则添加；若路由表中有，但有效位为 0，则删除后添加；若路由表中有，且有效位为 1，说明该段子网可通过更短的路径到达，则继续循环。

```

1 void *calculating_rtable_thread(void *param) {
2     while (1) {
3         .....
4         rt_entry_t *rt_entry = NULL;

```

```

5     list_for_each_entry(rt_entry, &rttable, list) {
6         if (0 == rt_entry->gw) rt_entry->valid = 1;
7         else rt_entry->valid = 0;
8     }
9     for (int t1 = 1; t1 < VNUM; t1++) {
10        u32 rid = v_list[t1];
11        u32 gw_rid = (0 == path[t1])? v_list[t1] : v_list[path[t1]];
12        iface_info_t *iface = NULL;
13        mospf_nbr_t *nbr = NULL;
14        list_for_each_entry(iface, &instance->iface_list, list) {
15            int found = 0;
16            ..... // To find the nbr whose rid equals gw_rid.
17            if (found) break;
18        }
19        mospf_db_entry_t *db_entry = NULL;
20        list_for_each_entry(db_entry, &mospf_db, list) {
21            if (db_entry->rid == rid) {
22                for(int t1 = 0; t1 < db_entry->nadv; t1++) {
23                    rt_entry_t *rt_entry = NULL;
24                    int found = 0;
25                    ..... // To find the rt_entry whose dest equals
26                        // db_entry->array[t1].subnet
27                    if (0 == found) {
28                        rt_entry_t *new = new_rt_entry(db_entry->array[t1].subnet, \
29                            db_entry->array[t1].mask, nbr->nbr_ip, iface);
30                        new->valid = 1;
31                        add_rt_entry(new);
32                    } else if (nbr->nbr_ip != rt_entry->gw && 0 == rt_entry->valid) {
33                        remove_rt_entry(rt_entry);
34                        rt_entry_t *new = new_rt_entry(db_entry->array[t1].subnet, \
35                            db_entry->array[t1].mask, nbr->nbr_ip, iface);
36                        new->valid = 1;
37                        add_rt_entry(new);
38                    } else rt_entry->valid = 1;
39                }
40            }
41        }
42        .....

```

```
43 }
```

2.2 数据库记录老化

给链路状态数据库结构增加保活时间域，每秒 +1。该域超过 35 则删除该条记录，同时将其对应的每个 lsa 信息都从路由表中找到响应条目并删除；但是将网关为 0，即从内核中读入的路由条目认为恒有效，不会被删除。如果子网仍然可达，则可通过下次计算路由表的过程中计算出一个新的路径。

```
1 mospf_db_entry_t *db_entry = NULL, *q;
2 list_for_each_entry_safe(db_entry, q, &mospf_db, list) {
3     db_entry->alive++;
4     if (db_entry->alive >= MOSPF_DATABASE_TIMEOUT) {
5         rt_entry_t *rt_entry = NULL;
6         rt_entry_t *q = longest_prefix_match(db_entry->rid);
7         u32 gw = q->gw;
8         list_for_each_entry_safe(rt_entry, q, &rtable, list) {
9             if(0 != gw && rt_entry->gw == gw) {
10                 fprintf(stdout, "DEBUG: remove rt_entry to %x.\n", rt_entry->dest);
11                 remove_rt_entry(rt_entry);
12             }
13         }
14         list_delete_entry(&db_entry->list);
15         free(db_entry->array);
16         free(db_entry);
17     }
18 }
```

3 实验结果及分析

```
"Node: h1"
root@sjy-PC:/media/sjy/新加坡/网[ ]/12-mospf_2# traceroute 10.0.6.22
traceroute to 10.0.6.22 (10.0.6.22), 30 hops max, 60 byte packets
 1 10.0.1.1 (10.0.1.1) 0.081 ms 0.035 ms 0.023 ms
 2 10.0.2.2 (10.0.2.2) 0.166 ms 0.167 ms 0.163 ms
 3 10.0.4.4 (10.0.4.4) 1.002 ms 0.995 ms 0.987 ms
 4 10.0.6.22 (10.0.6.22) 1.035 ms 1.028 ms 1.039 ms
root@sjy-PC:/media/sjy/新加坡/网[ ]/12-mospf_2# traceroute 10.0.6.22
traceroute to 10.0.6.22 (10.0.6.22), 30 hops max, 60 byte packets
 1 10.0.1.1 (10.0.1.1) 0.674 ms 1.277 ms 1.271 ms
 2 10.0.3.3 (10.0.3.3) 1.338 ms 1.337 ms 1.338 ms
 3 10.0.5.4 (10.0.5.4) 1.420 ms 1.425 ms 1.419 ms
 4 10.0.6.22 (10.0.6.22) 2.124 ms 2.160 ms 2.175 ms
root@sjy-PC:/media/sjy/新加坡/网[ ]/12-mospf_2#

mininet> xterm r1 r2 r3 r4 h1
mininet> link r2 r4 down
mininet> [ ]
```

可以看出结果是正确的。