Отчёт по лабораторной работе «Динамическая IP-маршрутизация»

Гребенюк Александр Андреевич

9 декабря 2015 г.

Содержание

1. Настройка сети

1.1. Топология сети

Топология сети и используемые IP-адреса показаны на рисунке 1.
Перечень узлов, на которых используется динамическая IP-маршрутизация:

– **r1-5**,

– **wsp1**

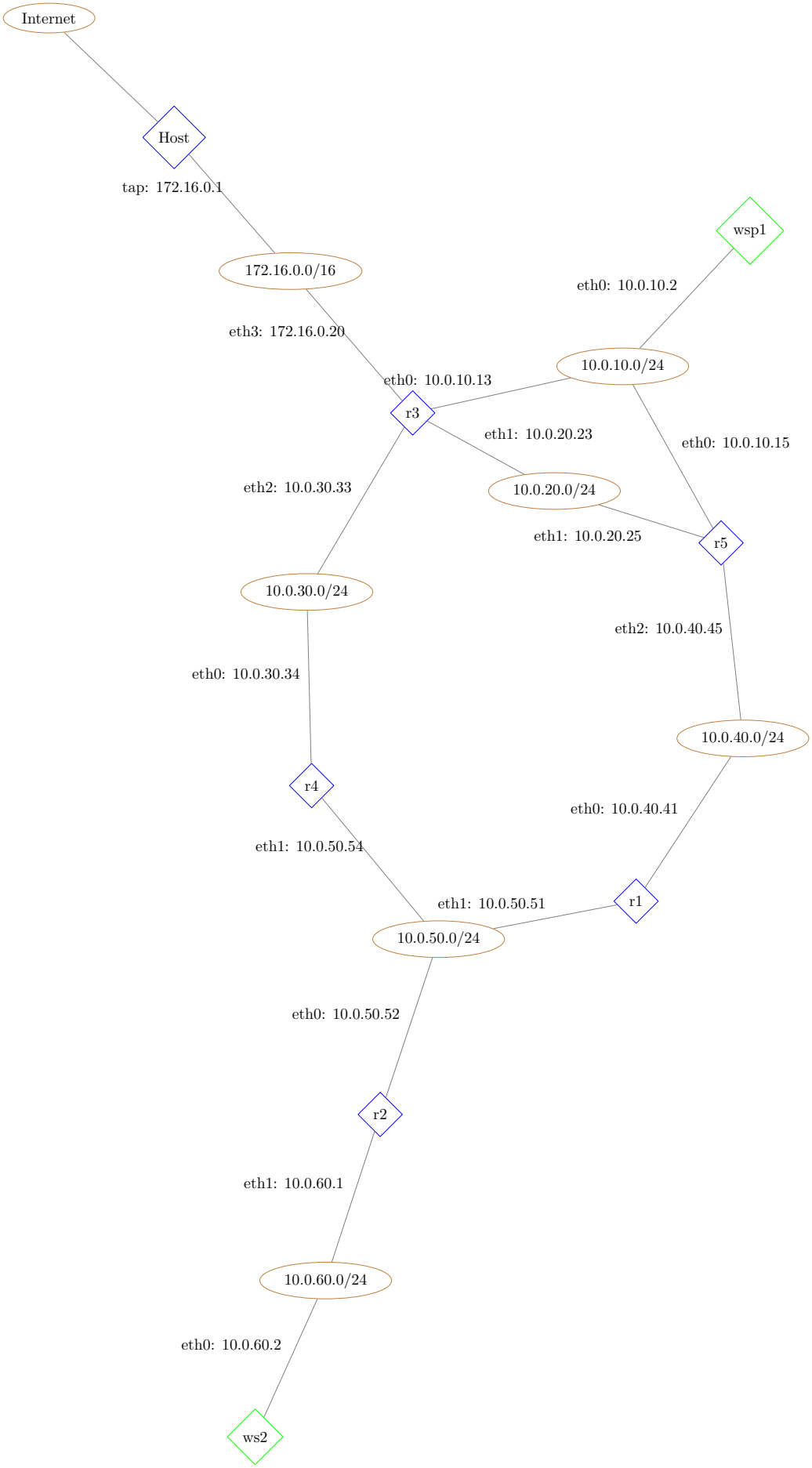

Рис. 1: Топология сети

2. Назначение IP-адресов

– Ниже приведён файл настройки протокола IP маршрутизатора **r1**.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
    address 10.0.40.41
    netmask 255.255.255.0

auto eth1
iface eth1 inet static
    address 10.0.50.51
    netmask 255.255.255.0
```

– Ниже приведён файл настройки протокола IP маршрутизатора **r2**.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
    address 10.0.50.52
    netmask 255.255.255.0

auto eth1
iface eth1 inet static
    address 10.0.60.1
    netmask 255.255.255.0
```

– Ниже приведён файл настройки протокола IP маршрутизатора **r3**.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
    address 10.0.10.13
    netmask 255.255.255.0

auto eth1
iface eth1 inet static
    address 10.0.20.23
    netmask 255.255.255.0

auto eth2
iface eth2 inet static
```

```
    address 10.0.30.33
    netmask 255.255.255.0
```

– Ниже приведён файл настройки протокола IP маршрутизатора **r4**.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
    address 10.0.30.34
    netmask 255.255.255.0

auto eth1
iface eth1 inet static
    address 10.0.50.54
    netmask 255.255.255.0
```

– Ниже приведён файл настройки протокола IP маршрутизатора **r5**.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 10.0.10.15
netmask 255.255.255.0

auto eth1
iface eth1 inet static
address 10.0.20.25
netmask 255.255.255.0

auto eth2
iface eth2 inet static
address 10.0.40.45
netmask 255.255.255.0
```

– Ниже приведён файл настройки протокола IP рабочей станции **ws2**.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 10.0.60.2
netmask 255.255.255.0
gateway 10.0.60.1
```

– Ниже приведён файл настройки протокола IP рабочей станции **wsp1**.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 10.0.10.2
netmask 255.255.255.0
gateway 10.0.10.13
```

2.1. Настройка протокола RIP

– Ниже приведен файл /**etc**/**quagga**/**ripd.conf** маршрутизатора **r1**.

```
! Этот настройки, касающиеся протокола RIP.
router rip

! Раскомментируйте ниже все интерфейсы, подключённые
! к сетям с другими маршрутизаторами.
network eth0
network eth1
! network eth2

! Уменьшаем значения всех таймеров для ускорения опытов.
! Рассылка: 10 сек., устаревание: 60 сек., сборка мусора: 120 сек.
timers basic 10 60 120

! Следующие две строчки заставляют маршрутизатор
! добавлять в сообщения протокола RIP все известные ему маршруты.
redistribute kernel
redistribute connected

! Это имя файла журнала службы RIP.
! Его содержимое можно изучить в случае неполадок
log file /var/log/quagga/ripd.log
```

– Ниже приведен файл /**etc**/**quagga**/**ripd.conf** маршрутизатора **r2**.

```
! Этот настройки, касающиеся протокола RIP.
router rip

! Раскомментируйте ниже все интерфейсы, подключённые
! к сетям с другими маршрутизаторами.
network eth0
! network eth1
! network eth2
```

```
! Уменьшаем значения всех таймеров для ускорения опытов.
! Рассылка: 10 сек., устаревание: 60 сек., сборка мусора: 120 сек.
timers basic 10 60 120

! Следующие две строчки заставляют маршрутизатор
! добавлять в сообщения протокола RIP все известные ему маршруты.
redistribute kernel
redistribute connected

! Это имя файла журнала службы RIP.
! Его содержимое можно изучить в случае неполадок
log file /var/log/quagga/ripd.log
```

– Ниже приведен файл /**etc**/**quagga**/**ripd.conf** маршрутизатора **r3**.

```
! Этот настройки, касающиеся протокола RIP.
router rip

! Раскомментируйте ниже все интерфейсы, подключённые
! к сетям с другими маршрутизаторами.
network eth0
network eth1
network eth2

! Уменьшаем значения всех таймеров для ускорения опытов.
! Рассылка: 10 сек., устаревание: 60 сек., сборка мусора: 120 сек.
timers basic 10 60 120

! Следующие две строчки заставляют маршрутизатор
! добавлять в сообщения протокола RIP все известные ему маршруты.
redistribute kernel
! redistribute connected

! Это имя файла журнала службы RIP.
! Его содержимое можно изучить в случае неполадок
log file /var/log/quagga/ripd.log
```

– Ниже приведен файл /**etc**/**quagga**/**ripd.conf** маршрутизатора **r4**.

```
! Этот настройки, касающиеся протокола RIP.
router rip

! Раскомментируйте ниже все интерфейсы, подключённые
! к сетям с другими маршрутизаторами.
network eth0
network eth1
! network eth2
```

```
! Уменьшаем значения всех таймеров для ускорения опытов.
! Рассылка: 10 сек., устаревание: 60 сек., сборка мусора: 120 сек.
timers basic 10 60 120

! Следующие две строчки заставляют маршрутизатор
! добавлять в сообщения протокола RIP все известные ему маршруты.
redistribute kernel
redistribute connected

! Это имя файла журнала службы RIP.
! Его содержимое можно изучить в случае неполадок
log file /var/log/quagga/ripd.log
```

– Ниже приведен файл /**etc**/**quagga**/**ripd.conf** маршрутизатора **r5**.

```
! Этот настройки, касающиеся протокола RIP.
router rip

! Раскомментируйте ниже все интерфейсы, подключённые
! к сетям с другими маршрутизаторами.
network eth0
network eth1
network eth2

! Уменьшаем значения всех таймеров для ускорения опытов.
! Рассылка: 10 сек., устаревание: 60 сек., сборка мусора: 120 сек.
timers basic 10 60 120

! Следующие две строчки заставляют маршрутизатор
! добавлять в сообщения протокола RIP все известные ему маршруты.
redistribute kernel
redistribute connected

! Это имя файла журнала службы RIP.
! Его содержимое можно изучить в случае неполадок
log file /var/log/quagga/ripd.log
```

Ниже приведен файл /**etc**/**quagga**/**ripd.conf** рабочий станции, связанной с несколькими маршрутизаторами **wsp1**.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 10.0.10.15
netmask 255.255.255.0
```

```
auto eth1
iface eth1 inet static
address 10.0.20.25
netmask 255.255.255.0

auto eth2
iface eth2 inet static
address 10.0.40.45
netmask 255.255.255.0
```

3. Проверка настройки протокола RIP

Вывод **traceroute** от **wsp1** до **ws2** при нормальной работе сети.

```
root@wsp1:~# traceroute 10.0.60.2
traceroute to 10.0.60.2 (10.0.60.2), 30 hops max, 60 byte packets
 1  10.0.10.15 (10.0.10.15)  0.156 ms  0.058 ms  0.056 ms
 2  10.0.40.41 (10.0.40.41)  0.179 ms  0.122 ms  0.114 ms
 3  10.0.50.52 (10.0.50.52)  0.212 ms  0.158 ms  0.129 ms
 4  10.0.60.2 (10.0.60.2)  0.263 ms  0.185 ms  0.196 ms
```

Вывод **traceroute** от узла **wsp1** до внешнего IP **8.8.8.8**

```
root@wsp1:~# traceroute 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1  10.0.10.13 (10.0.10.13)  0.293 ms  0.123 ms  0.091 ms
 2  172.16.0.1 (172.16.0.1)  0.358 ms  0.157 ms  0.213 ms
 3  192.168.1.1 (192.168.1.1)  1.691 ms  2.834 ms  2.674 ms
 4  78.107.125.197 (78.107.125.197)  6.329 ms  6.208 ms  6.108 ms
 5  85.21.0.236 (85.21.0.236)  5.998 ms  5.896 ms  5.764 ms
 6  10.2.254.78 (10.2.254.78)  8.914 ms  9.846 ms  9.739 ms
 7  195.14.54.228 (195.14.54.228)  9.645 ms  9.525 ms  9.420 ms
 8  195.14.54.141 (195.14.54.141)  9.282 ms  8.734 ms  8.643 ms
 9  78.107.184.43 (78.107.184.43)  9.642 ms 195.14.54.79 (195.14.54.79)
↪  8.868 ms  8.739 ms
10  195.14.32.22 (195.14.32.22)  57.939 ms 72.14.221.110 (72.14.221.110)
↪  7.814 ms  7.712 ms
11  216.239.47.127 (216.239.47.127)  7.618 ms 216.239.47.141
↪  (216.239.47.141)  6.069 ms  9.735 ms
12  8.8.8.8 (8.8.8.8)  9.594 ms  9.447 ms  9.342 ms
```

Модифицированный udp-клиент для перехвата RIP-сообщений:

```
/* Перехватчик RIP-трафика */

#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
```

```c
#include <netinet/in.h>
#include <arpa/inet.h>
#include <string.h>
#include <time.h>
#include <stdio.h>
#include <unistd.h>
#include <errno.h>

/* размер буфера для входящих датаграмм */
#define BUFSIZE 512
#define PORT 520

struct rip_route {
    uint8_t code;
    uint8_t addr_family;
    uint16_t tag;
    struct in_addr addr;
    struct in_addr mask;
    struct in_addr nexthop;
    uint32_t metric;
};

/* вычисление размера CIDR-маски */
int mask_len(struct in_addr mask) {
    int len = 0;
    int bit_pos = 0;
    while ((bit_pos < 32) && (mask.s_addr & (1 << bit_pos))) {
        len++; bit_pos++;
    }
    return len;
}

/* вывод в консоль содержания RIP-сообщения
 * (сообщение занимает первые `len` байт в буфере)
 */
void dump_rip(char *buffer, int len) {
    struct rip_route *rt = (struct rip_route *)(buffer + 4);
    while ((char*) rt < buffer + len) {
        printf("\tAFI %d addr %s/%d, tag 0x%04x, metric %u",
            rt->addr_family, inet_ntoa(rt->addr), mask_len(rt->mask),
            rt->tag, ntohl(rt->metric));
        printf(",  nexthop: %s\n", inet_ntoa(rt->nexthop));
        rt++;
    }
}

void die(const char* msg)
```

```
{
    perror("bind");
    exit(1);
}

int main(void) {
    struct sockaddr_in addr;     /* для своего адреса */
    //struct sockaddr_in sender;  /* для адреса отправителя */
    socklen_t addrlen;           /* размер структуры с адресом */
    int sk;                      /* файловый дескриптор сокета */
    char buffer[BUFSIZE+1];      /* буфер для приёма датаграмм */
    //ssize_t size;                /* размер полученных данных */
    int optvalue;                /* значение параметра сокета */
    struct ip_mreqn membership; /* информация о multicast-группе */


    /* TODO:
        создать сокет,
        подключиться к multicast-группе,
        связать сокет с портом,
        начать приём сообщений в бесконечном цикле,
        принятые сообщения передавать в функцию dump_rip(buffer, len)
    */

    if ((sk = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) < 0)
        die("create udp socket");

    /* разрешаем повторное использование адресов */
    optvalue = 1;
    if (setsockopt(sk, SOL_SOCKET, SO_REUSEADDR, (void *)&optvalue,
↪  sizeof(optvalue)) < 0)
        die("reuse addr");

    memset(&addr, 0, sizeof(addr));
    addr.sin_family = AF_INET;
    addr.sin_port = htons(520);
    addr.sin_addr.s_addr = htonl(INADDR_ANY);
    if (bind(sk, (struct sockaddr*)&addr, sizeof(addr)) < 0)
        die("bind");

    /* вступаем в multicast-группу */
    memset(&membership, 0, sizeof(membership));
    membership.imr_multiaddr.s_addr = inet_addr("224.0.0.9");
    membership.imr_address.s_addr = INADDR_ANY;
    membership.imr_ifindex = 2; /* индекс сетевого интерфейса eth0 (см. `ip
↪  a`) */
```

```
    if (setsockopt(sk, IPPROTO_IP, IP_ADD_MEMBERSHIP, (void *)&membership,
↪ sizeof(membership)) < 0)
        die(''multicast membership'');

    while (1)
    {
        int rlen = 0;
        if ((rlen = recvfrom(sk, buffer, BUFSIZE, 0, (struct sockaddr
↪ *)&addr, &addrlen)) < 0)
            die(''recvfrom'');
        dump_rip(buffer, rlen);
    }

    return 0;
}
```

Перехваченный обмен сообщениями:

```
root@r5:~/ripcatch# ./ripcatch32
        AFI 2 addr 10.0.20.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.30.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.50.0/24, tag 0x0000, metric 2,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.60.0/24, tag 0x0000, metric 3,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.10.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.30.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.40.0/24, tag 0x0000, metric 2,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.50.0/24, tag 0x0000, metric 2,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.60.0/24, tag 0x0000, metric 3,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.20.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.40.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.50.0/24, tag 0x0000, metric 2,  nexthop: 0.0.0.0
```

Таблица RIP:

```
     Network            Next Hop         Metric From            Tag Time
C(i) 10.0.10.0/24       0.0.0.0               1 self              0
C(i) 10.0.20.0/24       0.0.0.0               1 self              0
R(n) 10.0.30.0/24       10.0.10.13            2 10.0.10.13        0 00:53
C(i) 10.0.40.0/24       0.0.0.0               1 self              0
R(n) 10.0.50.0/24       10.0.40.41            2 10.0.40.41        0 00:59
R(n) 10.0.60.0/24       10.0.40.41            3 10.0.40.41        0 00:59
```

Таблица маршрутизации:

```
root@r5:~/ripcatch# ip r
10.0.10.0/24 dev eth0  proto kernel  scope link  src 10.0.10.15
10.0.20.0/24 dev eth1  proto kernel  scope link  src 10.0.20.25
10.0.30.0/24 via 10.0.10.13 dev eth0  proto zebra  metric 2
10.0.40.0/24 dev eth2  proto kernel  scope link  src 10.0.40.45
10.0.50.0/24 via 10.0.40.41 dev eth2  proto zebra  metric 2
10.0.60.0/24 via 10.0.40.41 dev eth2  proto zebra  metric 3
```

4. Расщепленный горизонт и испорченные обратные обновления

Расщепленный горизонт + испорченные обратные обновления

```
root@r5:~/ripcatch# ./ripcatch32
        AFI 2 addr 10.0.30.0/24, tag 0x0000, metric 2,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.50.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.60.0/24, tag 0x0000, metric 2,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.10.0/24, tag 0x0000, metric 16,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.20.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.30.0/24, tag 0x0000, metric 16,  nexthop: 10.0.10.13
```

Расщепленный горизонт - испорченные обратные обновления

```
root@r5:~/ripcatch# ./ripcatch32
        AFI 2 addr 10.0.20.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.30.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.50.0/24, tag 0x0000, metric 2,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.60.0/24, tag 0x0000, metric 3,  nexthop: 0.0.0.0
        AFI 0 addr 0.0.0.0/0, tag 0x0000, metric 16,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.20.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.40.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.20.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.30.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.50.0/24, tag 0x0000, metric 2,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.60.0/24, tag 0x0000, metric 3,  nexthop: 0.0.0.0
        AFI 0 addr 0.0.0.0/0, tag 0x0000, metric 16,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.10.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
```

Расщепленный горизонт отключен

```
root@r5:~/ripcatch# ./ripcatch32
        AFI 2 addr 10.0.20.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.30.0/24, tag 0x0000, metric 1,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.50.0/24, tag 0x0000, metric 2,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.60.0/24, tag 0x0000, metric 3,  nexthop: 0.0.0.0
        AFI 0 addr 0.0.0.0/0, tag 0x0000, metric 16,  nexthop: 0.0.0.0
        AFI 0 addr 0.0.0.0/0, tag 0x0000, metric 16,  nexthop: 0.0.0.0
        AFI 0 addr 0.0.0.0/0, tag 0x0000, metric 16,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.30.0/24, tag 0x0000, metric 2,  nexthop: 10.0.10.13
```

5. Имитация устранимой поломки в сети

Отключаем **r5**

Вывод таблицы RIP непосредственно перед истечением таймера устаревания (на маршрутизаторе **r1**).

```
     Network           Next Hop         Metric From            Tag Time
R(n) 10.0.10.0/24      10.0.40.45            2 10.0.40.45        0 00:55
R(n) 10.0.20.0/24      10.0.40.45            2 10.0.40.45        0 00:55
R(n) 10.0.30.0/24      10.0.50.54            2 10.0.50.54        0 00:51
C(i) 10.0.40.0/24      0.0.0.0               1 self              0
C(i) 10.0.50.0/24      0.0.0.0               1 self              0
R(n) 10.0.60.0/24      10.0.50.52            2 10.0.50.52        0 00:50
```

Перестроенная таблица на этом же маршрутизаторе

```
     Network           Next Hop         Metric From            Tag Time
R(n) 0.0.0.0/0         10.0.50.54            3 10.0.50.54        0 00:57
R(n) 10.0.10.0/24      10.0.50.54            3 10.0.50.54        0 00:57
R(n) 10.0.20.0/24      10.0.50.54            3 10.0.50.54        0 00:57
R(n) 10.0.30.0/24      10.0.50.54            2 10.0.50.54        0 00:57
C(i) 10.0.40.0/24      0.0.0.0               1 self              0
C(i) 10.0.50.0/24      0.0.0.0               1 self              0
R(n) 10.0.60.0/24      10.0.50.52            2 10.0.50.52        0 00:56
```

Вывод **traceroute** от **ws1** до **r1** до отключения **r5**

```
root@wsp1:~# traceroute 10.0.40.41
traceroute to 10.0.40.41 (10.0.40.41), 30 hops max, 60 byte packets
 1  10.0.10.15 (10.0.10.15)  0.172 ms  0.056 ms  0.058 ms
 2  10.0.40.41 (10.0.40.41)  0.165 ms  0.103 ms  0.129 ms
```

Вывод **traceroute** от **ws1** до **r1** после отключения **r5**

```
root@wsp1:~# traceroute 10.0.40.41
traceroute to 10.0.40.41 (10.0.40.41), 30 hops max, 60 byte packets
 1  10.0.10.13 (10.0.10.13)  0.119 ms  0.060 ms  0.057 ms
 2  10.0.30.34 (10.0.30.34)  0.165 ms  0.149 ms  0.100 ms
 3  10.0.40.41 (10.0.40.41)  0.232 ms  0.183 ms  0.144 ms
```

6. Имитация неустранимой поломки в сети

Отключаем **r2**

Таблица на маршрутизаторе **r5**

```
     Network           Next Hop         Metric From            Tag Time
C(i) 10.0.10.0/24      0.0.0.0               1 self              0
C(i) 10.0.20.0/24      0.0.0.0               1 self              0
R(n) 10.0.30.0/24      10.0.10.13            2 10.0.10.13        0 00:58
C(i) 10.0.40.0/24      0.0.0.0               1 self              0
R(n) 10.0.50.0/24      10.0.40.41            2 10.0.40.41        0 00:55
R(n) 10.0.60.0/24      10.0.40.41           16 10.0.40.41        0 01:15
```

Перехват RIP-сообщений на маршрутизаторе **r5**

```
root@r5:~/ripcatch# cat ripcatch.log | grep 16
        AFI 2 addr 10.0.60.0/24, tag 0x0000, metric 16,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.60.0/24, tag 0x0000, metric 16,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.60.0/24, tag 0x0000, metric 16,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.60.0/24, tag 0x0000, metric 16,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.60.0/24, tag 0x0000, metric 16,  nexthop: 0.0.0.0
        AFI 2 addr 10.0.60.0/24, tag 0x0000, metric 16,  nexthop: 0.0.0.0
```