СВД
Встраиваемые
Системы

Настройка и управление IPSec VPN в ЗОСРВ «Нейтрино»

Чуриков Эдуард

Техник-программист
Группа сетевых технологий

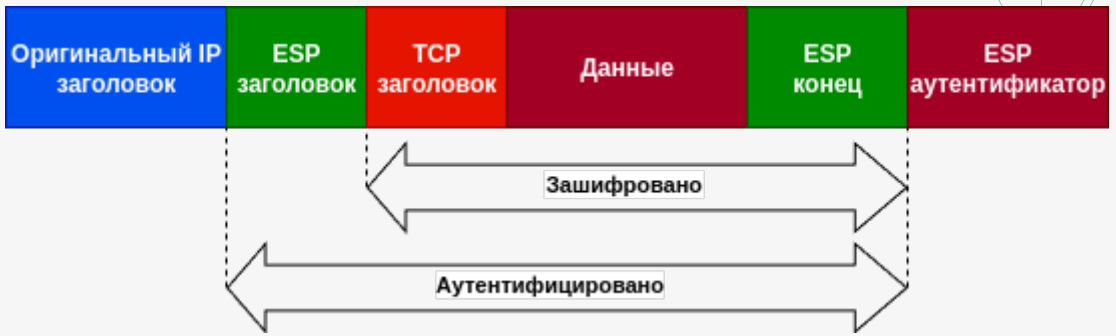
Общий обзор набора технологий IPSec

- **Протокол IKEv2 (RFC 7296)**
  - Инициализация соединения (выбор алгоритмов, обмен DH)
  - Аутентификация (PSK, EAP, X.509)
  - Создание Security associations (SA)
- **Протокол ESP (RFC 4303)**
  - Шифрование полезной нагрузки (AES, 3DES и др.)
  - Защита целостности (HMAC-SHA2, AES-XCBC-MAC и др)
  - Работает в транспортном и туннельном режимах
- **Протокол AH (RFC 4302)**
  - Аутентификация и целостность данных, без шифрования
  - Добавляет заголовок с HMAC (MD5, SHA1, SHA2)
  - Работает в транспортном и туннельных режимах. На практике применяется редко
- **Технология NAT-T (RFC 3948)**
  - Позволяет IPSec работать через NAT-маршрутизаторы
  - Инкапсулирует ESP в UDP (4500 порт)

Режимы работы IPSec

Транспортный режим

- Шифруется/подписывается только полезная нагрузка IP-пакета (данные TCP/UDP, ICMP и т. д.), а заголовок IP остаётся без изменений
- IP-адреса видны маршрутизаторам
- Защита точка-точка, между двумя хостами



Туннельный режим

- Шифруется/подписывается весь IP-пакет (включая IP заголовок). Исходный IP-пакет инкапсулируется внутрь нового IP-пакета с новым заголовком.
- Внутренние IP-адреса скрыты, можно построить „виртуальную“ сеть.
- Классический VPN

Демоны IPSec: от racoon к charon

Критерий	racoon (ipsec-tools)	charon (strongSwan)
Протокол IKE	IKEv1	IKEv1, IKEv2 (с MOBIKE протоколом)
Поддержка NAT-T	Отсутствует	Полная
Поддержка сертификатов	Ограниченная (X.509 без EAP, PKI)	Полная X.509 с PKI, EAP
Основное использование	Простые VPN	Современные VPN: H-to-H, S-to-S, Roadwarrior
Протоколы шифрования	Устаревшие. Новые не внедряются	Современные и легко расширяемые
Поддержка реализации IPSec	Ограниченная. Не развивается	Активно обновляется

Состав утилит и сервисов strongSwan

- **charon — основной IKE демон**
  - Содержит в себе реализацию IKEv1, IKEv2 с MOBIKE протоколом.
  - Занимается установкой IKE SA, CHILD SA, обменом ключами
  - Взаимодействует с сетевым стеком через PF_KEY сокет

- **swanctl — утилита для управления демоном**
  - Загрузка конфигурационных файлов
  - Инициализация соединения
  - Просмотр активных соединений

- **pki — утилита для работы с ключами и сертификатами**
  - Генерация RSA/ECC ключей
  - Выпуск самоподписанных и промежуточных сертификатов
  - Управление цепочками X.509

Состав конфигурационных файлов

```
├── strongswan.conf - Общий файл конфигурации плагинов, демона, утилит
├── strongswan.d - Каталог с модульными конфигурациями
│   ├── charon - Каталог с конфигурациями плагинов демона
│   │   ├── socket-default.conf
│   │   ├── sshkey.conf
│   │   ├── updown.conf
│   │   ├── vici.conf
│   │   └── ...
│   ├── charon.conf - Конфигурация демона
│   ├── charon-logging.conf - Конфигурация логирования демона
│   ├── pki.conf - Конфигурация утилиты pki
│   └── swanctl.conf - Конфигурация утилиты swanctl
└── swanctl - Каталог с учётными данными
    ├── private/
    ├── pubkey/
    ├── rsa/
    ├── .../
    ├── swanctl.conf - Конфигурация соединений, секретов и пулов IP-адресов
```

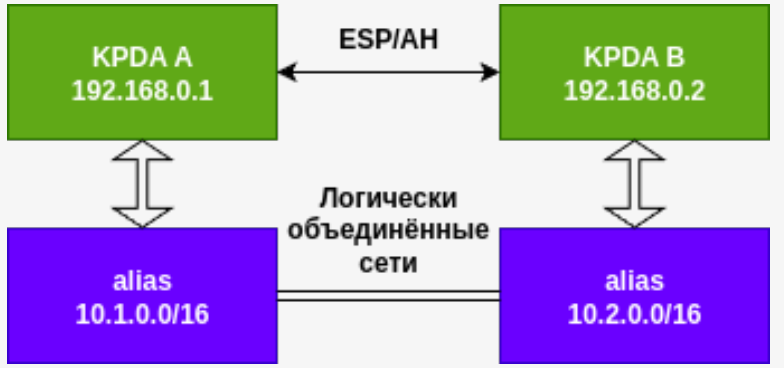
Конфигурации strongSwan

Host-to-Host

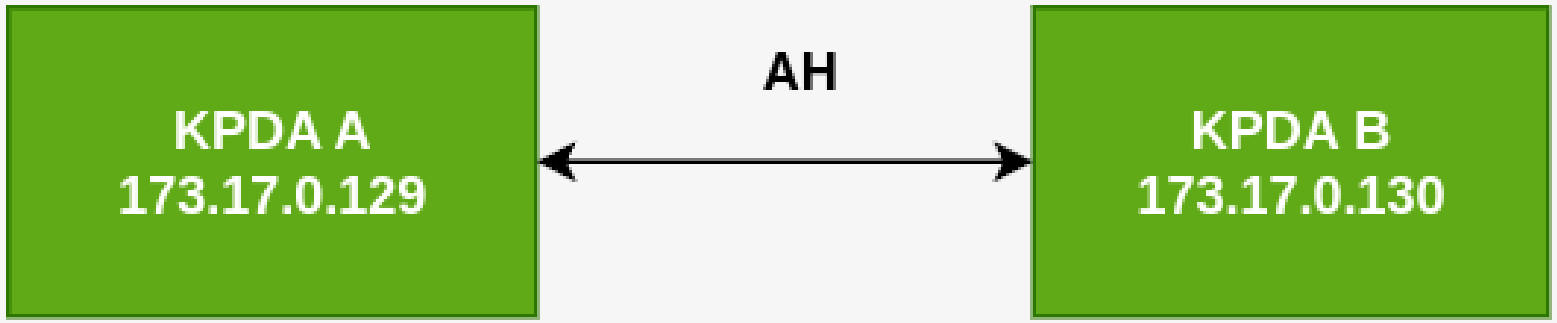
- Конфигурация между двумя хостами, не имеющих внутренней сети
- Поддерживается транспортным режимом
- Поддерживается туннельным режимом (с ограничениями)

Site-to-Site

- Конфигурация при которой две отдельные сети объединяются в одну логическую (классический VPN)
- Поддерживается туннельным режимом
- Поддерживается транспортным режимом (с ограничениями)



Host-to-Host (транспортный режим)



- **Описание**
  - В данной схеме между двумя узлами настраивается защищённое соединение с использованием протокола AH (Authentication Header), обеспечивающего аутентификацию и контроль целостности IP-пакетов.
  - Может применяться там, где важна гарантия подлинности и целостности данных, но нет требований к сокрытию содержимого (например, в закрытых сетях).

Host-to-Host (транспортный режим)

- **KPDA A**
  - swanctl.conf
- **Описание**
  - Соединение между двумя хостами
  - Аутентификация по паролю
  - В данном случае используется сочетание протокола AH и транспортного режима → данные не шифруются, заголовок не подменяется

```
connections { # Блок соединений
    ah-h2h { # Имя соединения (туннеля)
        version = 2 # Используем IKEv2
        local_addrs  = 173.17.0.129 # Локальный IP адрес для туннеля
        remote_addrs = 173.17.0.130 # Удалённый IP адрес (peer)

        local { # Настройки локальной стороны
            auth = psk # Аутентификация через Pre-Shared Key
            id = kpda-a # Идентификатор локальной стороны
        }
        remote { # Настройки удалённой стороны
            auth = psk # Аутентификация через Pre-Shared Key
            id = kpda-b # Идентификатор удалённой стороны
        }

        children { # Дочерние SA (Security Associations)
            ah-sa { # Имя SA
                local_ts  = 173.17.0.129/32  # Локальный Traffic Selector (IP/подсеть)
                remote_ts = 173.17.0.130/32  # Удалённый Traffic Selector
                mode = transport # Режим IPsec: transport (шифруются только данные пакета)
                ah_proposals = aes128-sha256-modp2048 # Алгоритмы аутентификации и DH группа
            }
        }
    }
}

secrets { # Блок секретов для аутентификации
    ike-kpda { # Имя секрета
        id-1 = kpda-a # Идентификатор первой стороны
        id-2 = kpda-b # Идентификатор второй стороны
        secret = "mysecret" # Общий секрет
    }
}
# Include config snippets
include conf.d/*.conf
```

Host-to-Host (транспортный режим)

- **KPDA B**
  - swanctl.conf
- **Описание**
  - Конфигурация должна быть зеркальной. Поменялись только local-remote параметры

```
connections {
    ah-h2h {
        version = 2
        local_addrs  = 173.17.0.130
        remote_addrs = 173.17.0.129

        local {
            auth = psk
            id = kpda-b
        }
        remote {
            auth = psk
            id = kpda-a
        }

        children {
            ah-sa {
                local_ts  = 173.17.0.130/32
                remote_ts = 173.17.0.129/32
                mode = transport
                ah_proposals = aes128-sha256-modp2048
            }
        }
    }
}

secrets {
    ike-kpda {
        id-1 = kpda-a
        id-2 = kpda-b
        secret = "mysecret"
    }
}
# Include config snippets
include conf.d/*.conf
```

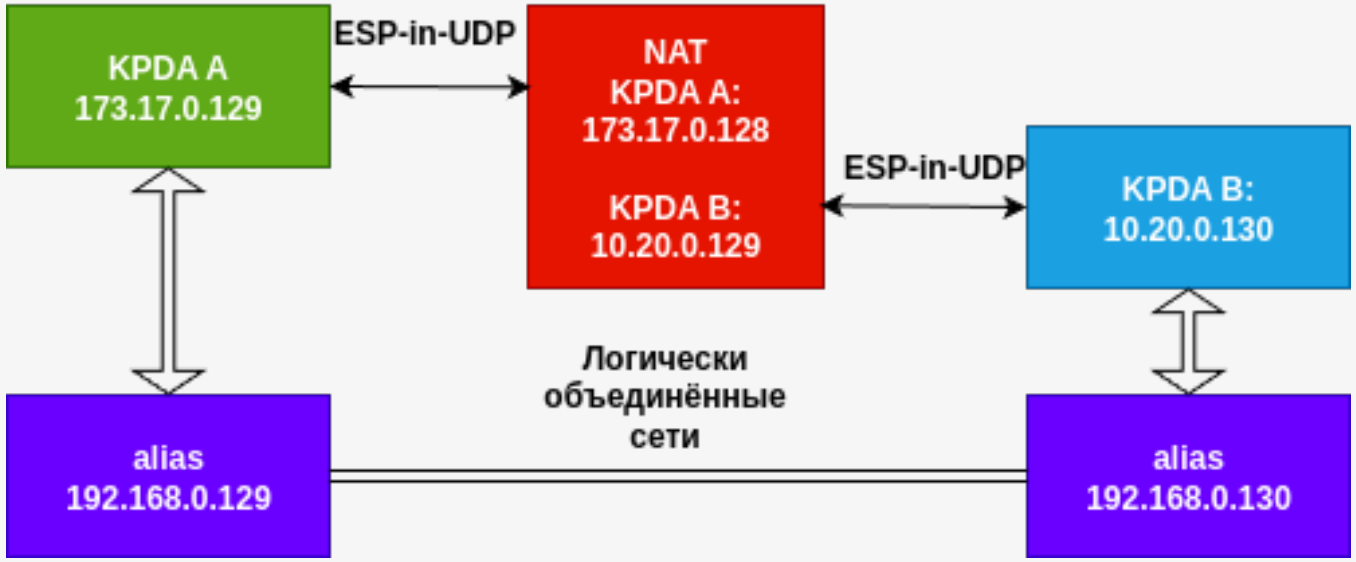
Host-to-Host (транспортный режим)

- **Результат — трафик аутентифицирован**
  - Уверенны, что пакет пришёл от заявленного отправителя
  - Пакет не был изменён в пути
  - Содержимое не зашифровано, но целостно и аутентично.

- **При использовании ping**
  - Видим протокол AH и его заголовок

```
▸ Frame 8475: 122 bytes on wire (976 bits), 122 bytes captured (976 bits) on interface vmnet2, id 0
▸ Ethernet II, Src: VMware_9b:29:75 (00:0c:29:9b:29:75), Dst: VMware_72:a5:98 (00:0c:29:72:a5:98)
▸ Internet Protocol Version 4, Src: 173.17.0.129, Dst: 173.17.0.130
▾ Authentication Header
    Next header: ICMP (1)
    Length: 4 (24 bytes)
    Reserved: 0000
    AH SPI: 0xcb23d0c8
    AH Sequence: 2
    AH ICV: b34178f70f876d7547b0cde8
▸ Internet Control Message Protocol
```

Site-to-Site с NAT-T (туннельный режим)



- **Описание**
  - В данной схеме между узлами KPDA A и KPDA B устанавливается защищённое соединение через устройство NAT. Для работы ESP протокола через NAT используется технология NAT-T, которая инкапсулирует ESP пакеты в UDP
  - Внутренние сети (192.168.0.129 и 192.168.0.130) логически объединены, что позволяет обеспечить защищённый обмен данными между двумя площадками так, как будто они находятся в одной локальной сети.

Site-to-Site с NAT-T (туннельный режим)

- **KPDA A**
  - swanctl.conf
- **Описание**
  - Соединение между двумя внутренними сетями.
  - Используется NAT-T, указаны внешние IP-адреса машин и внутренние для VPN
  - Аутентификация по паролю
  - В данном случае используется сочетание протокола ESP и туннельного режима → IP-заголовок подменяется, данные шифруются

```
connections { # Блок соединений
  net-b { # Имя соединения
    local_addrs = 173.17.0.129 # Локальный IP адрес для установления соединения
    remote_addrs = %any # Разрешаем подключение от любого удалённого IP

    local { # Настройки локальной стороны
      auth = psk # Аутентификация через Pre-Shared Key
      id = 173.17.0.129 # Идентификатор локальной стороны
    }
    remote {
      auth = psk # Аутентификация через Pre-Shared Key
      id = 10.20.0.130 # Идентификатор удалённой стороны
    }
    children {
      net-b-child { # Дочерние SA (Security Associations)
        local_ts = 192.168.0.129/32 # Локальный Traffic Selector (IP/подсеть)
        remote_ts = 192.168.0.130/32 # Удалённый Traffic Selector (IP/подсеть)
        esp_proposals = aes128-sha1-modp2048 # Алгоритмы ESP: AES128 (шифрование), SHA1 (аутентификация), MODP2048 (DH группа)
        start_action = none # Ручной старт соединения
      }
    }
    version = 2 # IKEv2
    mobike = yes # MOBIKE протокол (поддержка смены IP адреса без разрыва соединения)
    reauth_time = 3600 # Время (сек) до повторной аутентификации (реавторизации) IKE SA
    rekey_time = 1800 # Время (сек) до пересоздания ключей (pekey) для CHILD SA
    encap = yes # Принудительное включение UDP-инкапсуляции
  }
}

secrets { # Блок секретов (ключей) для аутентификации
  ike-net-b {
    id = 173.17.0.129
    secret = "supersecretkey"
  }
  ike-net-b {
    id = 10.20.0.130
    secret = "supersecretkey"
  }
}
# Include config snippets
include conf.d/*.conf
```

Site-to-Site с NAT-T (туннельный режим)

- **KPDA B**
  - swanctl.conf
- **Описание**
  - Конфигурация должна быть зеркальной. Поменялись local-remote параметры
  - Добавлена автоматическая инициация соединения при загрузке конфига при помощи **swanctl**

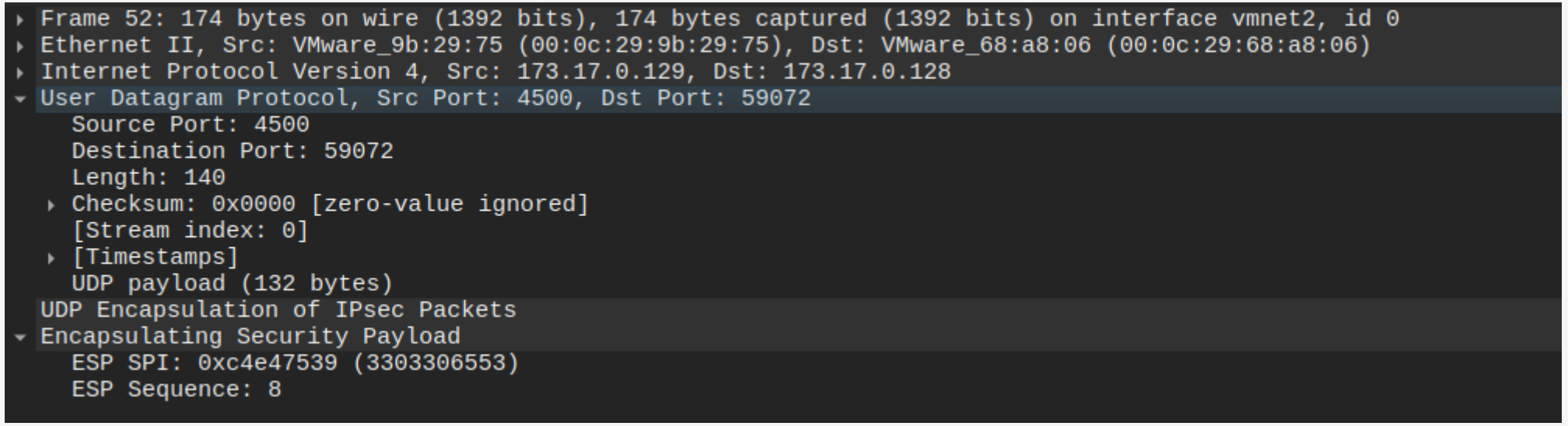
```
connections {
  net-a {
    local_addrs = 10.20.0.130
    remote_addrs = 173.17.0.129

    local {
      auth = psk
      id = 10.20.0.130
    }
    remote {
      auth = psk
      id = 173.17.0.129
    }
    children {
      net-a-child {
        local_ts = 192.168.0.130/32
        remote_ts = 192.168.0.129/32
        esp_proposals = aes128-sha1-modp2048
        start_action = start    # KPDA B инициирует туннель!
      }
    }
    version = 2
    mobike = yes
    reauth_time = 3600
    rekey_time = 1800
    encap = yes
  }
}

secrets {
  ike-net-b {
    id = 173.17.0.129
    secret = "supersecretkey"
  }
  ike-net-b {
    id = 10.20.0.130
    secret = "supersecretkey"
  }
}
# Include config snippets
include conf.d/*.conf
```

Site-to-Site с NAT-T (туннельный режим)

- **Важное замечание**
  - Из-за добавления дополнительных заголовков в ESP-in-UDP протоколе, необходимо уменьшить MTU на интерфейсе до 1400 байт
- **Результат — трафик зашифрован**
  - ESP трафик инкапсулирован в UDP
  - Наличие NAT — не проблема
  - Содержимое скрыто, внутренние IP-адреса скрыты
- **При использовании ping**
  - Видим протокол ESP-in-UDP и его заголовок



СВД
Встраиваемые
Системы

Спасибо за внимание!

Чуриков Эдуард

Техник-программист
Группа сетевых технологий

ул. Кузнецовская, д. 19,
г. Санкт-Петербург
+7 (812) 317-55-56
www.kpda.ru
support@kpda.ru

СВД
Встраиваемые
Системы

Инструменты IP-маршрутизации в ЗОСРВ “Нейтрино”

Щукин Александр

Руководитель группы сетевых технологий

Содержание мастер-класса

- Типы маршрутизации
- Протоколы динамической маршрутизации
- Настройка резервируемого канала
- Настройка динамической маршрутизации в сложной сети

Типы маршрутизации

- **Статическая маршрутизация**

  Ручная настройка маршрутов, применяется в малых сетях с неизменной топологией. Совместно с другими сервисами статическая маршрутизация может быть использована для обеспечения резервного канала обмена данными.

- **Динамическая маршрутизация**

  Автоматическая настройка маршрутов сервисами, реализующими протоколы обмена маршрутной информацией. Применяется для сложных сетей с изменяющейся топологией.

Протоколы динамической маршрутизации

- **Протокол RIPv1 (Routing Information Protocol version 1)**

  Протокол обмена маршрутной информацией. Данные передаются с помощью UDP-датаграмм по широковещательному адресу. Маска подсети не передаётся, RIPv1 использует классовую адресацию сети. Размер настраиваемой сети ограничен 15 пересылками (hop) пакета.

- **Протокол RIPv2 (Routing Information Protocol version 2)**

  Улучшенная версия RIPv1. Обмен данными может происходить с помощью группового адреса 224.0.0.9 (уменьшение нагрузки на узлы, которые не прослушивают данный групповой адрес). Для предотвращения произвольной и/или неверной настройки таблиц маршрутизации может быть использована аутентификация сообщений с открытым паролем либо с md5-подписью.

- **Протокол IRDP (ICMP Router Discovery Protocol)**

  Протокол используется узлами для обнаружения маршрутизаторов в сети. Маршрутизатор периодически рассылает по групповому адресу сообщение «Router Advertisment». А узел сети при включении может запросить информацию о маршрутизаторе с помощью запроса «Router Solicitation».

Настройка резервируемого канала

Постановка задачи

Требуется настроить резервируемый канал с помощью сервиса отслеживания состояния интерфейсов ifwatchd и статической маршрутизации.

Используемые инструменты

- Сервис ifwatchd

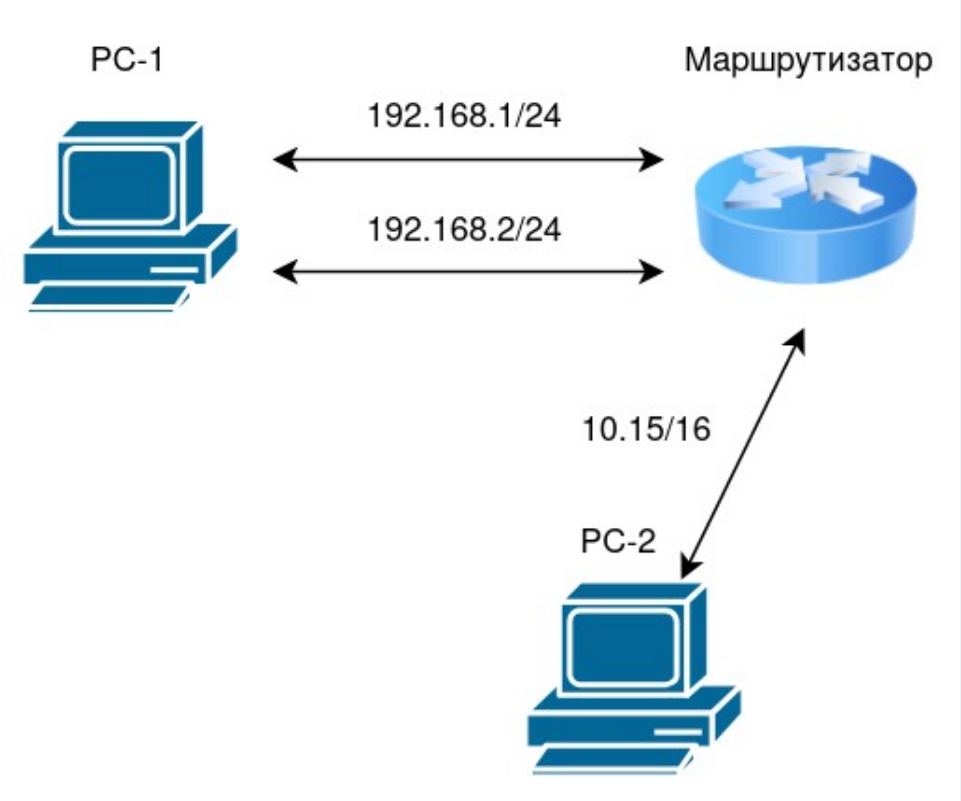
  С помощью сокета маршрутизации может реагировать на изменения адресов интерфейсов, на создание интерфейсов и на изменение состояния соединения.

- Утилита route

  Используется для ручного управления маршрутами. Также позволяет вывести таблицу маршрутизации и в режиме мониторинга показывать активность сервисов динамической маршрутизации.

Настройка резервного канала

- **Конфигурация сети**
  - PC-1 (ЗОСРВ Нейтрино)
    - Интерфейс wm0: 192.168.1.30/24
    - Интерфейс wm1: 192.168.2.30/24
  - Маршрутизатор (ЗОСРВ Нейтрино)
    - Интерфейс wm0: 192.168.1.1/24
    - Интерфейс wm1: 192.168.2.1/24
    - Интерфейс wm2: 10.15.1.1/16
  - PC-2 (ЗОСРВ Нейтрино)
    - Интерфейс wm0: 10.15.1.10/16



Настройка резервного канала

Настройка PC-1

Для смены статических маршрутов сервису ifwatchd требуются скрипты обработки появления (картинка слева) и потери (картинка справа) связи.

```
#!/bin/sh

REMOTE_NET="10.15"
IFACE_SET=("wm0" "wm1")
GW_SET=("192.168.1.1" "192.168.2.1")

if [ $1 = "wm0" ]; then
    GW=${GW_SET[0]}
    RESERV_IFACE=${IFACE_SET[1]}
    RESERV_GW=${GW_SET[1]}
elif [ $1 = "wm1" ]; then
    GW=${GW_SET[1]}
    RESERV_IFACE=${IFACE_SET[0]}
    RESERV_GW=${GW_SET[0]}
fi

out=$(/sbin/route show | /usr/bin/grep "$REMOTE_NET")
if [ -z "$out" ]; then
    /sbin/route add -net $REMOTE_NET $GW
else
    out=$(ifconfig $RESERV_IFACE | grep "status.*active")
    if [ -z "$out" ]; then
        /sbin/route delete -net $REMOTE_NET
        /sbin/route add -net $REMOTE_NET $GW
    fi
fi
```

```
#!/bin/sh

REMOTE_NET="10.15"
IFACE_SET=("wm0" "wm1")
GW_SET=("192.168.1.1" "192.168.2.1")

if [ $1 = "wm0" ]; then
    GW=${GW_SET[0]}
    RESERV_IFACE=${IFACE_SET[1]}
    RESERV_GW=${GW_SET[1]}
elif [ $1 = "wm1" ]; then
    GW=${GW_SET[1]}
    RESERV_IFACE=${IFACE_SET[0]}
    RESERV_GW=${GW_SET[0]}
fi

out=$(/sbin/route show | /usr/bin/grep "$REMOTE_NET.*$GW")
if ! [ -z "$out" ]; then
    /sbin/route delete -net $REMOTE_NET
    out=$(ifconfig $RESERV_IFACE | grep "status.*active")
    if ! [ -z "$out" ]; then
        /sbin/route add -net $REMOTE_NET $RESERV_GW
    fi
fi
```

Настройка резервного канала

Настройка PC-1

Скриптам ifwatchd перед запуском требуется установить права на исполнение.

```
# chmod +x /etc/ifwatchd.d/ifwatchd-up
# chmod +x /etc/ifwatchd.d/ifwatchd-down
# ifwatchd -c /etc/ifwatchd.d/ifwatchd-up \
-n /etc/ifwatchd.d/ifwatchd-down wm0 wm1
```

Настройка маршрутизатора

На маршрутизаторе необходимо включить пересылку пакетов. Это можно сделать либо при запуске сетевого менеджера io-pkt-ksz

```
# io-pkt-ksz -p tcpip forward
```

либо после с помощью команды

```
# sysctl -w net.inet.ip.forwarding=1
```

Настройка резервного канала

Настройка PC-2

Для ответа пакетам с PC-1 на PC-2 требуется настроить обратный путь. Для этого можно либо задать путь к обеим подсетям PC-1

```
# route add -net 192.168.1 10.15.1.1
# route add -net 192.168.2 10.15.1.1
```

либо установить путь по умолчанию к маршрутизатору

```
# route add default 10.15.1.1
```

Настройка резервного канала

Проверка работы

Проверим доступность PC-2 с PC-1 с помощью ping и посмотрим маршруты с помощью route. Маршрут к 10.15/16 проходит через 192.168.1.1 (картинка слева).

Разорвём соединение в подсети 192.168.1/24, проверим доступность PC-2 с PC-1 и маршруты после разрыва соединения в подсети 192.168.1/24. Маршрут изменился (картинка справа).

```
# ping -c1 10.15.1.10
PING 10.15.1.10 (10.15.1.10): 56 data bytes
64 bytes from 10.15.1.10: icmp_seq=0 ttl=63 time=3 ms

----10.15.1.10 PING Statistics----
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 3/3/3 ms   variance = 0 ms^2
# route show
Routing tables

Internet:
Destination        Gateway           Flags
10.15.0.0/16       192.168.1.1       UG
localhost.localdom 127.0.0.1         UH
192.168.1.0/24     link#2            U
192.168.1.1        02:03:0a:12:02:0b UHL
192.168.2.0/24     link#3            U
```

```
# ping -c1 10.15.1.10
PING 10.15.1.10 (10.15.1.10): 56 data bytes
64 bytes from 10.15.1.10: icmp_seq=0 ttl=63 time=2 ms

----10.15.1.10 PING Statistics----
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 2/2/2 ms   variance = 0 ms^2
# route show
Routing tables

Internet:
Destination        Gateway           Flags
10.15.0.0/16       192.168.2.1       UG
localhost.localdom 127.0.0.1         UH
192.168.1.0/24     link#2            U
192.168.1.1        02:03:0a:12:02:0b UHL
192.168.2.0/24     link#3            U
192.168.2.1        02:03:0a:12:02:0c UHL
```

Настройка динамической маршрутизации

Постановка задачи

Требуется настроить динамическую маршрутизацию в сети со сложной топологией с помощью сервисов, работающих согласно протоколу RIPv2 с аутентификацией сообщений md5-подписью.

Используемые инструменты

- Сервис routed

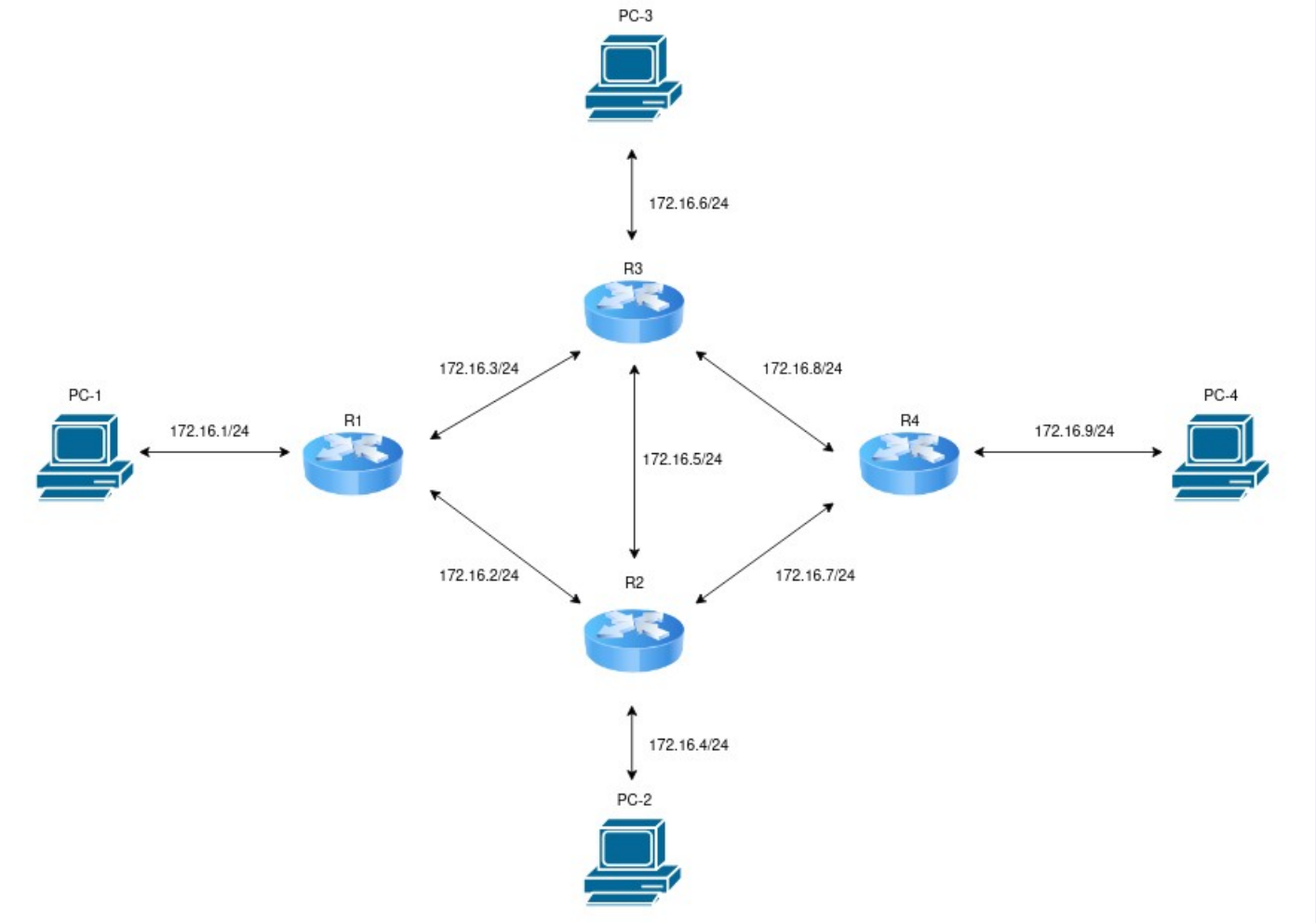
  Реализует управление маршрутизацией согласно RIPv1, RIPv2 и IRDP.
- Утилита netstat

  Расширенный просмотр информации о маршрутах.
- Утилита traceroute

  Вывод пути до узла.

Настройка динамической маршрутизации

- **Схема сети**



Настройка динамической маршрутизации

Конфигурация сети

- PC-1 (ЗОСРВ Нейтрино): 172.16.1.40/24
- PC-2 (ЗОСРВ Нейтрино): 172.16.4.42/24
- PC-3 (ЗОСРВ Нейтрино): 172.16.6.45/24
- PC-4 (ЗОСРВ Нейтрино): 172.16.9.47/24

- Маршрутизатор R1 (ЗОСРВ Нейтрино):
  - Интерфейс wm0: 172.16.1.41/24
  - Интерфейс wm1: 172.16.3.41/24
  - Интерфейс wm2: 172.16.2.41/24
- Маршрутизатор R2 (ЗОСРВ Нейтрино):
  - Интерфейс wm0: 172.16.2.43/24
  - Интерфейс wm1: 172.16.5.43/24
  - Интерфейс wm2: 172.16.7.43/24
  - Интерфейс wm3: 172.16.4.43/24
- Маршрутизатор R3 (ЗОСРВ Нейтрино):
  - Интерфейс wm0: 172.16.3.44/24
  - Интерфейс wm1: 172.16.6.44/24
  - Интерфейс wm2: 172.16.8.44/24
  - Интерфейс wm3: 172.16.5.44/24
- Маршрутизатор R4 (ЗОСРВ Нейтрино):
  - Интерфейс wm0: 172.16.8.46/24
  - Интерфейс wm1: 172.16.9.46/24
  - Интерфейс wm2: 172.16.7.46/24

Настройка динамической маршрутизации

Настройка PC-1 – PC-4

На хостах требуется определить путь по умолчанию. Для автоматизированного определения требуется запустить routed в пассивном режиме,

```
# routed -P no_rip &
```

таким образом routed будет использоваться только для прослушивания сообщений IRDP "Router Advertisment" и для отправки запросов IRDP "Router Solicitiation"

Настройка динамической маршрутизации

Настройка R1 - R4

Параметры функционирования routed на маршрутизаторах удобно задавать в стандартном конфигурационном файле routed - /etc/gateways. Настраиваемые опции:

- ripv2 – включить приём RIPv2.
- md5_passwd – незашифрованный пароль с идентификатором для аутентификации сообщений.
- ripv2_out – включить отправку RIPv2.

```
ripv2
md5_passwd=1ahsGbsfg234|1
ripv2_out
```

Конфигурационому файлу требуется настроить права и собственника. Для активации динамической маршрутизации остаётся настроить пересылку пакетов и запустить routed.

```
# chown root:root /etc/gateways
# chmod 600 /etc/gateways
# sysctl -w net.inet.ip.forwarding=1
# routed &
```

Настройка динамической маршрутизации

Проверка работы

На маршрутизаторе R1 выведем известные маршруты с помощью netstat.

```
# netstat -r
Routing tables

Internet:
Destination        Gateway            Flags    Refs      Use    Mtu  Interface
localhost.localdom localhost.localdom UH          0       24  33192  lo0
172.16.1/24        link#2             UC          1        0      -  wm0
172.16.1.40        02:03:0a:12:04:01  UHLc        0        0      -  wm0
172.16.2/24        link#4             UC          1        0      -  wm2
172.16.2.43        link#4             UHLc        2        0      -  wm2
172.16.3/24        link#3             UC          1        0      -  wm1
172.16.3.44        link#3             UHLc        3        0      -  wm1
172.16.4/23        172.16.2.43        UG          0        0      -  wm2
172.16.5/24        172.16.3.44        UG          0        0      -  wm1
172.16.6/24        172.16.3.44        UG          0        0      -  wm1
172.16.7/24        172.16.2.43        UG          0        0      -  wm2
172.16.8/23        172.16.3.44        UG          0        0      -  wm1
```

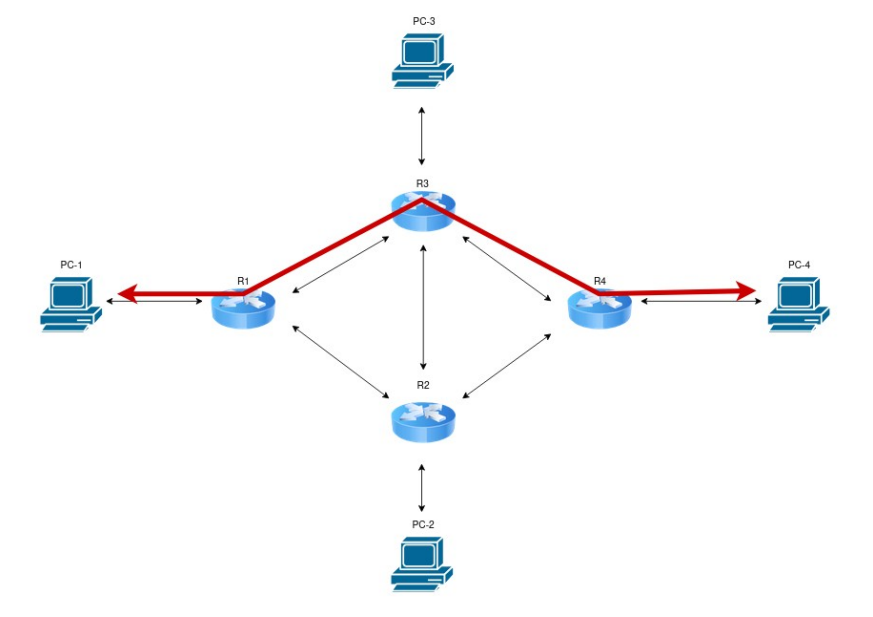
На PC-1 с помощью traceroute выведем путь до 172.16.9.47 (PC-4). Полученый маршрут: PC-1 <-> R1 <-> R3 <-> R4 <-> PC.

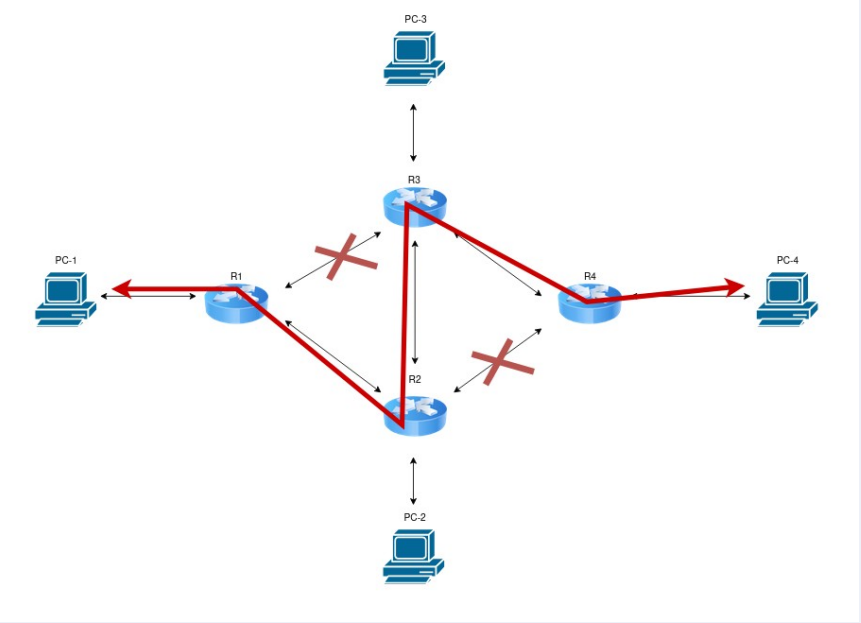
```
# traceroute 172.16.9.47
traceroute to 172.16.9.47 (172.16.9.47), 64 hops max, 40 byte packets
 1  172.16.1.41 (172.16.1.41)  1.000 ms  1.000 ms  1.000 ms
 2  172.16.3.44 (172.16.3.44)  2.000 ms  1.000 ms  1.000 ms
 3  172.16.8.46 (172.16.8.46)  3.000 ms  2.000 ms  2.000 ms
 4  172.16.9.47 (172.16.9.47)  3.000 ms  3.000 ms  3.000 ms
```

Настройка динамической маршрутизации

Проверка работы

В качестве проверки динамической маршрутизации разорвём соединение между R1 <-> R3 и R2 <-> R4 для того, чтоб получить маршрут: PC-1 <-> R1 <-> R2 <-> R3 <-> R4 <-> PC-4. На картинке слева красной линией изображён изначальный путь от PC-1 до PC-4, на картинке справа – искомый после отключения.





Настройка динамической маршрутизации

Проверка работы

Из-за особенностей устаревания записей в RIP перестройка маршрута может занять продолжительное время (от 3х минут). Проверим получившийся маршрут с помощью traceroute. Маршрут изменился.

```
# traceroute 172.16.9.47
traceroute to 172.16.9.47 (172.16.9.47), 64 hops max, 40 byte packets
 1  172.16.1.41 (172.16.1.41)  1.000 ms  1.000 ms  0.000 ms
 2  172.16.2.43 (172.16.2.43)  1.000 ms  1.000 ms  1.000 ms
 3  172.16.5.44 (172.16.5.44)  6.000 ms  2.000 ms  3.000 ms
 4  172.16.8.46 (172.16.8.46)  3.000 ms  3.000 ms  3.000 ms
 5  172.16.9.47 (172.16.9.47)  4.000 ms  3.000 ms  4.000 ms
```

На маршрутизаторе R1 выведем обновлённые маршруты. Ранее маршрут до подсети 172.16.9/24 (входит в 172.16.8/23) проходил через 172.16.3.44 (R3), теперь же он проходит через 172.16.2.43 (R2).

```
# netstat -r
Routing tables

Internet:
Destination        Gateway            Flags    Refs      Use    Mtu  Interface
localhost.localdom localhost.localdom UH          0      100  33192  lo0
172.16.1/24        link#2             UC          1        0      -  wm0
172.16.1.40        02:03:0a:12:04:01  UHLc        1       81      -  wm0
172.16.2/24        link#4             UC          1        0      -  wm2
172.16.2.43        02:03:0a:12:04:31  UHLc        2        0      -  wm2
172.16.3/24        link#3             UC          0        0      -  wm1
172.16.4/22        172.16.2.43        UG          0        0      -  wm2
172.16.8/23        172.16.2.43        UG          0       65      -  wm2
```

СВД
Встраиваемые
Системы

Спасибо за внимание!

Щукин Александр

Руководитель группы
сетевых технологий

+7 (812) 317-55-56
www.kpda.ru