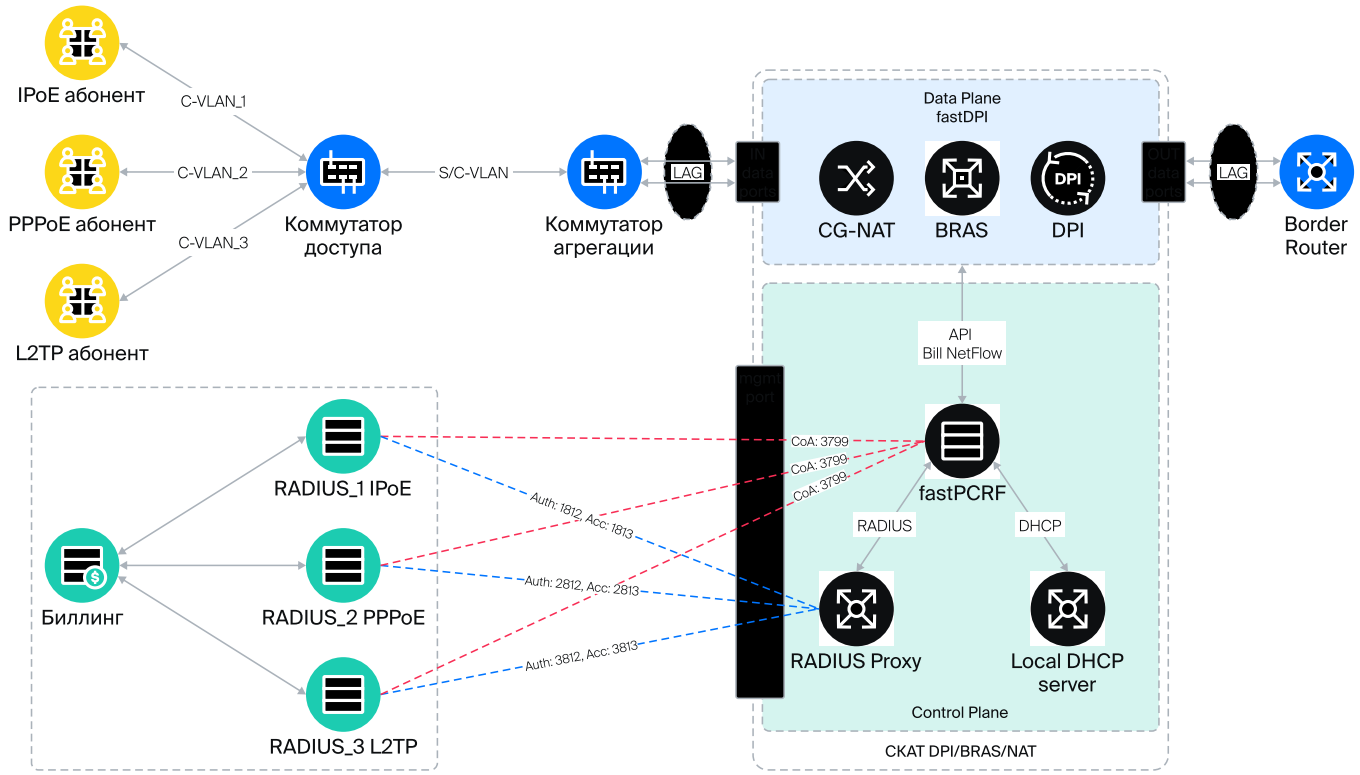
Содержание

Настройка балансировки и распределения по группам RADIUS серверов 3
Конфигурация FreeRADIUS как Proxy 3
Конфигурация виртуального сервера FreeRADIUS 6

Настройка балансировки и распределения по группам RADIUS серверов

Используется для сценария, когда необходимо распределить RADIUS трафик абонентов с разными типами доступа IPoE, PPPoE, L2TP между разными группами RADIUS серверов.



В данном примере рассмотрим распределение Авторизации и Аккаунтинга между группами серверов в режим load-balance для разных типов доступа между группами RADIUS серверов:

1. IPoE на RADIUS_1 (1812, 1813), RADIUS_1.1 (1822, 1823)
2. PPPoE на RADIUS_2 (2812, 2813), RADIUS_2.1 (2822, 2823)
3. L2TP на RADIUS_3 (3812, 3813), RADIUS_3.1 (3822, 3823)

Разделение Auth/Acc различных типов доступа происходит на основе атрибута VasExperts-Service-Type с использованием операторов ветвления (if-else).

RADIUS CoA сообщения могут отправляться от RADIUS серверов в BRAS (fastPCRF) следующим способом:

1. Напрямую от RADIUS (billing) в fastPCRF. Необходимо добавить Отдельные CoA-клиенты в fastPCRF.
2. Через Proxy: RADIUS (billing) → Proxy → fastPCRF. Описан пример ниже.

Конфигурация FreeRADIUS как Proxy

Конфигурация proxy расположена в `/etc/raddb/proxy.conf`. В ней определены основные

разделы:

```
proxy server{
  retry_delay = 5
  retry_count = 3
  default_fallback = no
  #dead_time = 120
  wake_all_if_all_dead = yes
}
```

- `retry_delay` — количество секунд ожидания после того, как сервер попытается установить соединение и завершится неудачей.
- `retry_count` — максимальное количество попыток отправки запроса к одному, прежде чем он будет считаться "мёртвым".
- `default_fallback` — параметр, отвечающий за ответ reject-ом на клиентский запрос при статусе всех серверов "мертвые".
- `wake_all_if_all_dead` — параметр, отвечающий за периодическую проверку состояния «мёртвых» серверов.

Раздел определяющий параметры "домашних" серверов, т.е. серверов на которых хранятся данные об абонентах.

```
home_server rad_1 {
        type = auth+acct
        ipaddr = 10.166.220.232
        port = 1812
        secret = secret
        # proto = udp
# optional items
        src_ipaddr = 10.16.20.117
        response_window = 6
        zombie_period = 40
        status_check = status-server
        check_interval = 6
        check_timeout = 4
        num_answers_to_alive = 2
        max_outstanding = 65536
        coa {
                irt = 2
                mrt = 16
                mrc = 5
                mrd = 30
        }
        limit {
              max_connections = 16
              max_requests = 0
              lifetime = 0
              idle_timeout = 0
        }
}
```

- `type` — для чего используется сервер, чаще всего это авторизация (`auth`), а также авторизация и аккаунтинг (`auth+acct`).
- `ipaddr` — адрес RADIUS сервера, также может быть `ipv6addr`.
- `port` — порт, на который проксировать запросы (обычно 1812). Также при режиме `auth+acct` назначается только первый порт для `auth`, а порт аккаунтинга определяется, как `port+1`.
- `proto` — протокол передачи данный, по умолчанию udp.
- `secret` — используется для "шифрования" и "подписи" пакетов между RADIUS и сервером проксирования.
- `src_ipaddr` — адрес, с которого отправляются запросы от прокси.
- `response_window` — период времени, в который прокси ожидает ответа на запрос от сервера. По истечению времени, сервер помечается как "zombie" и является наименее приоритетным для отправки.
- `zombie_period` — период времени, в который прокси ожидает ответ от сервера на любой пакет и по истечении которого помечает сервер статусом "мертвый".
- `status_check` — определяет, как проверять состояние сервера.
- `check_interval` — интервал между отправкой пакетов проверки состояния.
- `check_timeout` — время ожидания ответа на пакет проверки состояния.
- `num_answers_to_alive` — количество проверок состояния подряд, после которого сервер считается "живым".
- `max_outstanding` — количество пакетов, то есть разница между оправленными на сервер и полученными от сервера, при превышении которого пакеты перестают отправляться чтобы избежать перегрузки RADIUS сервера.
- `coa` — секция описания интервалов повторных перадач и их количества для Change of Authorization.
- `limit` — секция, актуальная только при указании протокола TCP. Это `max_connections` — максимальное кол-во подключений; `max_requests` — максимальное количество запросов в рамках одной сессии; `lifetime` — время жизни соединения в секундах; `idle_timeout` — максимальное время ожидания пакетов в рамках сессии, после истечения которого она будет закрыта. Значение `0` во всех параметрах данной секции означает "без ограничений".

Для proxy определите столько серверов, сколько необходимо. Далее необходимо сгруппировать их в pool, если отвечают за один и тот же тип авторизации. Также с помощью раздела `home_server_pool` осуществляется балансировка нагрузки и переключение между серверами.
В статье приведена конфигурация в единичном варианте, полная конфигурация доступна в архиве.

```
home_server_pool pool_rad_servers {
    type = load-balance
    home_server = rad_1
}
```

Важно, чтобы `home_server` были одного типа, то есть все `auth` или `auth+acct`.

Наконец, в разделе `realm` определяются некоторые параметры и указывается, какой пул серверов следует использовать для этой области.

```
realm rlm_prod_servers {
    pool = pool_rad_servers
```

```
    nostrip
}
```

При использовании пула только с серверами авторизации, можно использовать `auth_pool`, при пуле только с аккаунтингом `acct_pool`. Но лучше использовать просто `pool`, который включает оба предыдущих указателя.
Также сервер может проксировать пакеты CoA на основе атрибута `Operator-Name` через `coa_pool`. Для настройки CoA использовать `/etc/raddb/sites-available/coa`, в котором определяются параметры проксирования CoA-запросов. В файле конфигурации важно указать условия, при которых проксируется запрос в один из realm. А в каждом `home_server` определить параметры и для CoA (чаще всего они указаны по умолчанию).
Но наши рекомендации — для работы с CoA использовать прямую связку PCRF – RADIUS, так как все параметры задаются в `fastpcrf.conf` и общение происходит напрямую.
Для настройки этих функций есть статья.

И параметр `nostrip`, отвечающий за отсутствие деления User-Name.

Конфигурация виртуального сервера FreeRADIUS

Файл конфигурации расположен в `/etc/raddb/sites-available/default`. Все параметры указывать для `default`. Сначала раздел `listen`, в котором указать какие подсети и порты слушать, а также какие типы сообщений принимать.

```
server default {
listen {
        type = auth
        ipaddr = *
        port = 0
        limit {
              max_connections = 16
              lifetime = 0
              idle_timeout = 30
        }
}
listen {
        ipaddr = *
        port = 0
        type = acct
        limit {
        }
}
```

Также раздел для IPv6:

```
listen {
        type = auth
        ipv6addr = ::   # any.  ::1 == localhost
        port = 0
        limit {
```

```
                max_connections = 16
                lifetime = 0
                idle_timeout = 30
        }
}
listen {
        ipv6addr = ::
        port = 0
        type = acct
        limit {
        }
}
```

Далее раздел авторизации, в котором указаны доступные протоколы авторизации.
В этом разделе определено правило, по которому все запросы авторизации с атрибутом `VasExperts-Service-Type` `0` и `1` отправляются в область для DHCP, если атрибутом равен 2, 3 и 4 в область PPPoE, а остальные будут отброшены с Reject. Все значения атрибута для других типов авторизации с расшифровкой указаны в словаре vasexperts и в статье.

```
authorize {
        preprocess
        chap
        mschap
        digest
        suffix
        files
        -sql
        -ldap
        expiration
        logintime
        if (Tunnel-Type) {
            update control {
                Proxy-To-Realm := "rlm_prod_servers_3"
           }
        }

        else {
              if (VasExperts-Service-Type == 0  || VasExperts-Service-Type
== 1) {
          update control {
                Proxy-To-Realm := "rlm_prod_servers_1"
          }
        }
  else ( VasExperts-Service-Type == 2 || VasExperts-Service-Type == 3 ||
VasExperts-Service-Type == 4 ) {
                update control {
                      Proxy-To-Realm := "rlm_prod_servers_2"
                }
              }

              else {
```

```
                    reject
                }
        }
}
```

Раздел определяет методы аутентификации, которые будет использовать FreeRADIUS. Так как они пустые, то будут использоваться модули по умолчанию.

```
authenticate {
        Auth-Type PAP {
        }
        Auth-Type CHAP {
        }
        Auth-Type MS-CHAP {
        }
}

preacct {
        preprocess
        acct_unique
        suffix
        files
}
```

Раздел тарификации/аккаунтинга, в котором определены правила обработки пакетов с различными `VasExperts-Service-Type`, как и в модуле авторизации. Также важным отличием является первое правило для `Acct-Status-Type`, на основе которого прокси будет отправлять на RADIUS и общие пакеты включения тарификации, а не отбрасывать их.

```
accounting {
        -sql
        if (Acct-Status-Type == Accounting-On || Acct-Status-Type ==
Accounting-Off) {
            update control {
                Proxy-To-Realm := "rlm_acct_servers"
            }
          }

        else {
           if (Tunnel-Type) {
            update control {
                Proxy-To-Realm := "rlm_prod_servers_3"
            }
        }
        else {
              if (VasExperts-Service-Type == 0 || VasExperts-Service-Type ==
1) {
                update control {
                   Proxy-To-Realm := "rlm_prod_servers_1"
                }
              }
```

```
                else   (VasExperts-Service-Type == 2 || VasExperts-Service-
Type == 3 || VasExperts-Service-Type == 4 ) {
                    update control {
                      Proxy-To-Realm := "rlm_prod_servers_2"
                }
                }
            }
      }
}
session {
}
```

Завершающие разделы. Задается параметр таймаута сессии и действие системы при Reject.

```
post-auth {
        update reply {
                Session-Timeout := 4294967295
        }
        -sql
        Post-Auth-Type REJECT {
        }
        Post-Auth-Type Challenge {
        }
}
pre-proxy {
}
post-proxy {
}
}
```