

Отказоустойчивое решение (кластер) с одним интерфейсом. Аутентификация на сертификатах, СКЗИ «КриптоПро CSP 2.0»

Описание стенда

Сценарий иллюстрирует построение защищенного соединения между двумя подсетями, одна из которых - SN1-192.168.103.0/24 защищается шлюзом безопасности GW3, а вторая подсеть - SN2-10.0.110.0/24 защищается кластером из двух шлюзов безопасности GW1 и GW2. Устройства IPHost1 и IPHost2 смогут общаться между собой по защищенному каналу (VPN). Все остальные соединения разрешены, но защищаться не будут. В рамках данного сценария, для аутентификации, партнеры будут использовать сертификаты. В качестве криптопровайдера будет использован «КриптоПро CSP» версии 2.0 (Рисунок 1).

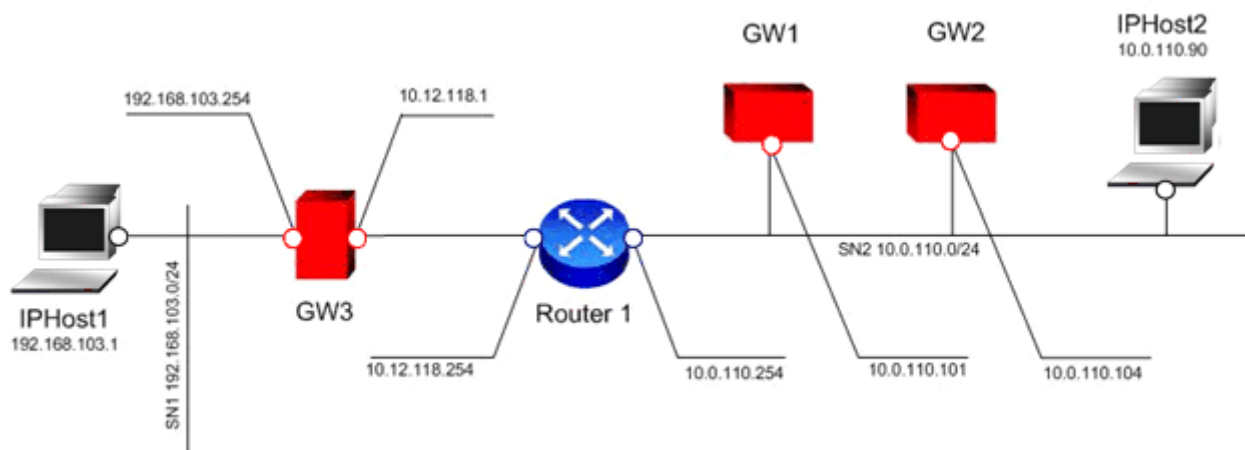


Рисунок 1

Параметры защищенного соединения

Параметры защищенного соединения между подсетями SN1 и SN2:

- Аутентификация на сертификатах
- IKE parameters:
 - Encryption algorithm – GOST
 - Hash algorithm – GOST
 - DH-group – group2 (1024)
- IPSec parameters:
 - ESP encryption algorithm – GOST

Логика работы кластера

В предложенном решении реализована следующая логика:

- В качестве кластера используются два шлюза безопасности CSP VPN Gate
- Шлюзы имеют различные роли – GW1 – основной шлюз безопасности, GW2 – резервный шлюз безопасности.
- Кроме шлюзов безопасности определено одно «надежное» устройство. На это устройство будут отправляться ICMP-пакеты для диагностики работоспособности сетевых интерфейсов шлюзов безопасности.
- В нормальном режиме весь трафик обрабатывается на основном шлюзе. Резервный шлюз в это время находится в режиме ожидания.
- В случае выхода из строя основного шлюза безопасности его заменяет резервный шлюз и обрабатывает весь проходящий трафик.
- После восстановления работоспособности основного шлюза безопасности резервный шлюз переходит в режим ожидания и отдает управление основному.
- Проверка работоспособности основного и резервного шлюзов безопасности, а также действия по активации и настройке интерфейсов производятся с помощью скриптов, устанавливаемых на шлюзы кластера.

Настройка шлюза безопасности GW1

Сначала настроим устройство GW1. Все настройки будем производить удаленно, получив доступ к устройству по протоколу ssh с правами суперпользователя.

Отключим посылку ICMP-сообщений о перенаправлении (ICMP-redirect), задав команду:

```
gw1#ndd -set /dev/ip ip_send_redirects 0
```

Если не отключить ICMP-redirect, то пакеты от IPHost2 будут направляться непосредственно на адрес Router1, минуя GW1.

Проверим, что ICMP-redirect отключен:

```
gw1#ndd /dev/ip ip_send_redirects
0
gw1#
```

Регистрация CA сертификата

Для регистрации CA сертификата выполним следующие действия:

1. Доставим файл CA сертификата на шлюз безопасности. Для доставки можно воспользоваться утилитой pscp.exe из пакета Putty. Файл ca.cer доставим в предварительно созданный каталог /certs:

```
pscp ca.cer root@10.0.110.101:/certs
```



Предупреждение

**Среда передачи в этом случае должна быть
доверенной**

2. С помощью утилиты `cert_mgr`, входящей в состав продукта, зарегистрируем сертификат в базе продукта:

```
gw1#/opt/VPNagent/bin/cert_mgr import -f /certs/ca.cer -t
```

Регистрация локального сертификата

Для регистрации локального сертификата в базе продукта выполним следующие действия:

1. Доставим в файловую систему шлюза безопасности файл локального сертификата. Снова воспользуемся утилитой `pscp.exe`.

```
pscp local1.cer root@10.0.110.101:/certs
```

2. Доставим контейнеры с секретными ключами для локального сертификата на шлюз безопасности.

```
pscp -r local1.000 root@10.0.110.101: /var/CPROcsp/users
```

3. Получим имя контейнера с секретными ключами, для этого используем утилиту `csptest`, входящую в состав ПО криптопровайдера:

```
gw1# /opt/CPROcsp/src/csptest/csptest -keyset -machinekeyset -verifycontext -enum_containers -unique
```

```
CryptAcquireContext succeeded.HCRYPTPROV: 1
                                HDIMAGE\\local1.000|                                local1
```

ОК.

4. Зарегистрируем локальный сертификат в базе продукта, используя утилиту `cert_mgr` из состава продукта. Импорт производим командой следующего вида:

```
gw1# /opt/VPNagent/bin/cert_mgr import -f /certs/local1.cer -kc 'HDIMAGE\\local1.000'
```

5. Убедимся, что сертификаты импортированы успешно:

```
gw1# /opt/VPNagent/bin/cert_mgr show
```

```
Found 2 certificates. No CRLs found.
```

```
1 Status: trusted 1.2.840.113549.1.9.1=support@s-terra.com,C=RU,ST=Moscow,L=Zelenograd,O=S-Terra CSP,OU=Support Group,CN=Support Group CA
```

```
2 Status: local 1.2.840.113549.1.9.1=support@s-terra.com,C=RU,ST=Moscow,L=Zelenograd,O=S-Terra CSP,OU=Support Group,CN=local1
```

Настройка адресов шлюза безопасности

Для предотвращения конфликтов IP-адресов рекомендуется настроить адреса, которые будут назначаться интерфейсам при старте ОС. Для установленного продукта CSP VPN Gate версии 2.2

и выше это можно выполнить при помощи скрипта `/usr/sbin/ipsetup`, входящего в состав продукта.

Для продукта CSP VPN Gate версии 2.1 такую настройку адресов можно выполнить в файле `/etc/hosts`.

Интерфейсу с именем `PHYS_1_1_NAME` должен быть назначен адрес `FIXED_1_1_PARAM`. Соответствие имен, адресов и конкретных значений смотрите в разделе [“Описание настраиваемых параметров основного скрипта устройства GW1”](#).

Установка скриптов

На шлюз безопасности GW1 установим два скрипта. Один скрипт будет отвечать за настройку интерфейса и оценку работоспособности шлюза, а второй скрипт будет отвечать за старт основного скрипта при перезагрузке операционной системы.

Основной скрипт

Основной скрипт сохраним в директории `/etc/init.d` под именем `vpn_main_gate` и выполним для него:

```
gw1#chmod a+x vpn_main_gate
```

Текст скрипта `vpn_main_gate`:

```
#!/bin/ksh

# Version 1.2

IP_RELIABLE1=10.0.110.254

PHYS_1_1_NAME=rtls0
TUNNEL_1_PARAM=10.0.110.101/24

TO_CHECK_1_NAME=${PHYS_1_1_NAME}:1
TO_CHECK_1_PARAM=10.0.110.102/24

FIXED_1_1_NAME=${PHYS_1_1_NAME}:2
FIXED_1_1_PARAM=10.0.110.103/24

CHECK_N=3
RELAX_TIMEOUT=4

PING=/usr/sbin/ping

do_ping1()
{
    $PING $IP_RELIABLE1 1 > /dev/null
}

```

```
notify()
{
  logger -p local7.notice "$1"
  echo `date` "$1" > /dev/console
}

on_fail()
{
  ifconfig $PHYS_1_1_NAME down 2> /dev/null
  ifconfig $TO_CHECK_1_NAME down 2> /dev/null
  ifconfig $FIXED_1_1_NAME $FIXED_1_1_PARAM broadcast + -deprecated 2>
/dev/null
  /opt/VPNagent/bin/lsp_mgr reload
notify 'Main gate is deactivated'
}

on_ok()
{
  ifconfig $TO_CHECK_1_NAME $TO_CHECK_1_PARAM broadcast + deprecated up 2>
/dev/null
  sleep $RELAX_TIMEOUT
  ifconfig $PHYS_1_1_NAME $TUNNEL_1_PARAM broadcast + up 2> /dev/null
  ifconfig $FIXED_1_1_NAME deprecated 2> /dev/null
  /opt/VPNagent/bin/lsp_mgr reload
notify 'Main gate is activated'
}

ifconfig $TO_CHECK_1_NAME plumb 2> /dev/null

ifconfig $FIXED_1_1_NAME plumb 2> /dev/null

ifconfig $PHYS_1_1_NAME $TUNNEL_1_PARAM down broadcast + 2> /dev/null

ifconfig $FIXED_1_1_NAME $FIXED_1_1_PARAM up broadcast + 2> /dev/null

if do_ping1; then
  RETRY_COUNT1=$CHECK_N
  RETRY_COUNT2=$CHECK_N
  on_ok
else
  RETRY_COUNT1=0
  RETRY_COUNT2=0
  on_fail
fi

while true; do
```

```

if [[ $RETRY_COUNT1 -gt 0 ]] && [[ $RETRY_COUNT2 -gt 0 ]]; then
    if do_ping1; then
        RETRY_COUNT1=$CHECK_N
    else
        RETRY_COUNT1=$((RETRY_COUNT1-1))
        if [ $RETRY_COUNT1 -eq 0 ]; then
            on_fail
        fi
    fi
fi

if do_ping1; then
    RETRY_COUNT2=$CHECK_N
else
    RETRY_COUNT2=$((RETRY_COUNT2-1))
    if [[ $RETRY_COUNT1 -gt 0 ]] && [[ $RETRY_COUNT2 -eq 0 ]]; then
        on_fail
    fi
fi

if [[ $RETRY_COUNT1 -gt 0 ]] && [[ $RETRY_COUNT2 -gt 0 ]]; then
    sleep 1
fi

else
    if do_ping1; then
        RETRY_COUNT1=$CHECK_N
        RETRY_COUNT2=$CHECK_N
        on_ok
    fi
fi

done

```

Описание настраиваемых параметров основного скрипта устройства GW1

В основном скрипте настроим следующие параметры:

IP_RELIABLE1 - IP-адрес надежного устройства.

PHYS_1_1_NAME - имя внешнего физического интерфейса.

TUNNEL_1_PARAM - параметры физического интерфейса: IP-адрес и длина префикса (количество единиц в сетевой маске).

TO_CHECK_1_NAME - имя виртуального интерфейса Virtual (to check), который будет использоваться для определения работоспособности шлюза безопасности (на него будет отправлять ping резервный шлюз безопасности).

TO_CHECK_1_PARAM - параметры виртуального интерфейса: IP-адрес и длина префикса (количество единиц в сетевой маске).

FIXED_1_1_NAME - имя виртуального интерфейса Virtual (fixed).

FIXED_1_1_PARAM - параметры виртуального интерфейса Virtual (fixed): IP-адрес и длина префикса (количество единиц в сетевой маске).

CHECK_N – количество неуспешных попыток ping, после которых считается, что связи нет. Чем меньше данное число, тем быстрее срабатывает переключение интерфейсов, но тем больше вероятность ложных срабатываний.

RELAX_TIMEOUT – таймаут (в секундах) между поднятием виртуальных интерфейсов Virtual (to check) и поднятием туннельного интерфейса. В этот таймаут резервный шлюз безопасности должен опустить туннельный интерфейс.

Запуск скрипта `vpn_main_gate` будет производиться с помощью вспомогательного скрипта при старте операционной системы. Запуск и остановку скрипта `vpn_main_gate` можно производить вручную, при помощи команд::

```
gw1#/etc/init.d/vpn_main_gate_init start
gw1#/etc/init.d/vpn_main_gate_init stop
```

Вспомогательный скрипт

Вспомогательный скрипт служит для обеспечения запуска и остановки основного скрипта.

Вспомогательный скрипт сохраним в директории `/etc/init.d` под именем `vpn_main_gate_init` и выполним для него

```
gw1#chmod a+x vpn_main_gate_init
```

Текст скрипта `vpn_main_gate_init`:

```
#!/bin/sh

# Version 1.2

VPN_MAIN_GATE_SCRIPT=/etc/init.d/vpn_main_gate
PS=/usr/bin/ps
GREP=/usr/bin/grep
KILL=/usr/bin/kill
AWK=/usr/bin/awk

case $1 in
'start')
    $VPN_MAIN_GATE_SCRIPT &
    ;;
'stop')
    PID=`$PS -ef | $GREP -v grep | $GREP $VPN_MAIN_GATE_SCRIPT | $AWK '{print $2}'`
    if [ ! -z "$PID" ] ; then
        /usr/bin/kill $PID 1> /dev/null 2>&1
    fi
    ;;
*)
    echo "Usage: vpn_main_gate_init { start | stop }"
    ;;
esac
```

Для того, чтобы скрипт `vpn_main_gate` вызывался при `Startup`-е и останавливался при `Shutdown`-е, сделаем линки на скрипт `vpn_main_gate_init`:

```
gw1#ln -s /etc/init.d/vpn_main_gate_init
/etc/rc2.d/S22vpn_main_gate_init

gw1#ln -s /etc/init.d/vpn_main_gate_init
/etc/rc0.d/K91vpn_main_gate_init
```

Создание политики безопасности

После регистрации сертификатов перейдем к созданию политики безопасности для GW1. Создавать политику будем в интерфейсе командной строки. Для входа в консоль перейдем в директорию `/opt/VPNagent/bin/` и запустим `cs_console`. Перейдем в режим настройки.

Зададим адрес шлюза по умолчанию:

```
gw1 (config)#ip route 0.0.0.0 0.0.0.0 10.0.110.254 1
```

Зададим параметры для IKE:

```
gw1 (config)#crypto isakmp policy 1
gw1 (config-isakmp)#hash md5
gw1 (config-isakmp)# encryption des
gw1 (config-isakmp)# authentication rsa-sig
gw1 (config-isakmp)# group 2
gw1 (config-isakmp)#exit
```

Создадим набор преобразований для IPsec:

```
gw1 (config)#crypto ipsec transform-set Gost esp-des
gw1 (cfg-crypto-trans)#mode tunnel
gw1 (cfg-crypto-trans)#exit
```

Опишем трафик, который планируется защищать. Для этого создадим расширенный список доступа.

```
gw1 (config)#ip access-list extended SN2toSN1
gw1 (config-ext-nacl)# permit ip 10.0.110.0 0.0.0.255 192.168.103.0
0.0.0.255
gw1 (config-ext-nacl)#exit
```

Создадим криптокарту:

```
gw1 (config)#crypto map CMAP 1 ipsec-isakmp
gw1 (config-crypto-map)# match address SN1toSN2
gw1 (config-crypto-map)# set transform-set Gost
```

```
gw1 (config-crypto-map)# set pfs group2
gw1 (config-crypto-map)# set peer 10.12.118.1
gw1 (config-crypto-map)#exit
```

«Привяжем» криптокарту к интерфейсу, на котором будет терминироваться туннель:

```
gw1 (config)#interface FastEthernet0/0
gw1 (config-if)#crypto map CMAP
gw1 (config-if)#exit
```

Отключим обработку списка отозванных сертификатов (CRL):

```
gw1 (config)#crypto ca trustpoint s-terra_technological_trustpoint
gw1 (ca-trustpoint)# crl optional
gw1 (ca-trustpoint)#exit
```

Настройка устройства GW1 завершена. При выходе из конфигурационного режима происходит загрузка конфигурации.

В Приложении представлен [текст cisco-like конфигурации](#) и [текст LSP, таблица маршрутов](#) и [вывод команды ifconfig -a](#). В выводе зафиксировано состояние интерфейсов для активного режима.

Настройка шлюза GW2

Отключим посылку ICMP-сообщений о перенаправлении (ICMP-redirect), задав команду:

```
gw2#ndd -set /dev/ip ip_send_redirects 0
```

Если не отключить ICMP-redirect, то пакеты от IPHost2 будут направляться непосредственно на адрес Router1, минуя GW2.

Проверим, что ICMP-redirect отключен:

```
gw2#ndd /dev/ip ip_send_redirects
0
gw2#
```

Регистрация CA и локального сертификатов производится так же, как и для шлюза безопасности GW1.

Настройка адресов шлюза безопасности

Для предотвращения конфликтов IP-адресов рекомендуется настроить адреса, которые будут назначаться интерфейсам при старте ОС. Для установленного продукта CSP VPN Gate версии 2.2 и выше это можно выполнить при помощи скрипта `/usr/sbin/ipsetup/`.

Для продукта CSP VPN Gate версии 2.1 такую настройку адресов можно выполнить в файле `/etc/hosts`.

Интерфейсу с именем `PHYS_2_1_NAME` должен быть назначен адрес `FIXED_2_1_PARAM`. Соответствие имен, адресов и конкретных значений смотрите в разделе [“Описание настраиваемых параметров основного скрипта устройства GW2”](#).

Установка скриптов

Шлюз безопасности GW2 в нашей схеме играет роль резервного шлюза. На нем также установим два скрипта – основной и вспомогательный.

Основной скрипт

Основной скрипт сохраним в директорию `/etc/init.d` под именем `vpn_reserve_gate` и выполним для него

```
gw2#chmod a+x vpn_reserve_gate
```

Текст скрипта `vpn_reserve_gate`:

```
#!/bin/ksh
# Version 1.2

IP_RELIABLE1=10.0.110.254

IP_TO_CHECK1=10.0.110.102

PHYS_2_1_NAME=iprb0

TUNNEL_1_PARAM=10.0.110.101/24

FIXED_2_1_NAME=${PHYS_2_1_NAME}:1
FIXED_2_1_PARAM=10.0.110.104/24

CHECK_N=3

PING=/usr/sbin/ping

notify()
{
    logger -p local7.notice "$1"
    echo `date` "$1" > /dev/console
}

do_ping1()
{
    $PING $IP_RELIABLE1 1 > /dev/null
}
```

```
do_ping_g1()
{
    $PING $IP_TO_CHECK1 1 > /dev/null
}
ifs_down()
{
    ifconfig $FIXED_2_1_NAME -deprecated 2> /dev/null
    ifconfig $PHYS_2_1_NAME down 2> /dev/null
    /opt/VPNagent/bin/lsp_mgr reload
notify 'Reserve gate is deactivated'
}

ifs_up()
{
    ifconfig $PHYS_2_1_NAME $TUNNEL_1_PARAM broadcast + up 2> /dev/null
    ifconfig $FIXED_2_1_NAME deprecated 2> /dev/null
    /opt/VPNagent/bin/lsp_mgr reload
notify 'Reserve gate is activated'
}

IFS_UP_FLAG=0

ifconfig $FIXED_2_1_NAME plumb 2> /dev/null
ifconfig $FIXED_2_1_NAME $FIXED_2_1_PARAM up broadcast + 2> /dev/null

ifconfig $PHYS_2_1_NAME $TUNNEL_1_PARAM down broadcast + 2> /dev/null

if do_ping_g1; then
    RETRY_COUNT_G1=$CHECK_N
    RETRY_COUNT_G2=$CHECK_N
    RETRY_COUNT1=$CHECK_N
    RETRY_COUNT2=$CHECK_N
    ifs_down
else
    RETRY_COUNT_G1=0
    RETRY_COUNT_G2=0

    if do_ping1; then
        RETRY_COUNT1=$CHECK_N
        RETRY_COUNT2=$CHECK_N
        ifs_up
        IFS_UP_FLAG=1
    else
```

```
        RETRY_COUNT1=0
        RETRY_COUNT2=0
        ifs_down
    fi
fi
while true; do
    CHECK_CONTINUE=0

    if [[ $RETRY_COUNT_G1 -gt 0 ]] && [[ $RETRY_COUNT_G2 -gt 0 ]]; then
        if do_ping_g1; then
            RETRY_COUNT_G1=$CHECK_N
            RETRY_COUNT1=$CHECK_N
        else
            RETRY_COUNT_G1=$((RETRY_COUNT_G1-1))
        fi

        if [[ $RETRY_COUNT_G1 -gt 0 ]] && [[ $RETRY_COUNT_G2 -gt 0 ]]; then
            sleep 1
        fi

    else
        if do_ping_g1; then
            RETRY_COUNT_G1=$CHECK_N
            RETRY_COUNT_G2=$CHECK_N
            RETRY_COUNT1=$CHECK_N
            RETRY_COUNT2=$CHECK_N
            ifs_down
            IFS_UP_FLAG=0
        fi
    fi

    if [[ $RETRY_COUNT_G1 -eq 0 ]] || [[ $RETRY_COUNT_G2 -eq 0 ]]; then
        if [[ $RETRY_COUNT1 -gt 0 ]] && [[ $RETRY_COUNT2 -gt 0 ]]; then
            if do_ping1; then
                RETRY_COUNT1=$CHECK_N
            else
                RETRY_COUNT1=$((RETRY_COUNT1-1))
                if [ $RETRY_COUNT1 -eq 0 ]; then
                    ifs_down
                    IFS_UP_FLAG=0
                fi
            fi
        fi
    fi
fi
```

```

        if [[ $RETRY_COUNT1 -eq $CHECK_N ]] && [[ $RETRY_COUNT2 -eq $CHECK_N
]] && [[ $IFS_UP_FLAG -eq 0 ]]; then
            ifs_up
            IFS_UP_FLAG=1
        fi

        if [[ $RETRY_COUNT1 -gt 0 ]] && [[ $RETRY_COUNT2 -gt 0 ]]; then
            sleep 1
        fi

    else
        if do_ping1 && [[ $IFS_UP_FLAG -eq 0 ]]; then
            RETRY_COUNT1=$CHECK_N
            RETRY_COUNT2=$CHECK_N
            ifs_up
            IFS_UP_FLAG=1
        fi
    fi
fi
done

```

Описание настраиваемых параметров основного скрипта устройства GW2

В основном скрипте настройте следующие параметры:

IP_RELIABLE1 - IP адрес надежного устройства.

IP_TO_CHECK1 - адрес интерфейса основного шлюза безопасности, по которому производится проверка его работоспособности.

PHYS_2_1_NAME - имя физического интерфейса резервного шлюза безопасности, на котором установлен туннельный адрес (общий для обоих шлюзов безопасности).

TUNNEL_1_PARAM - параметры физического интерфейса резервного шлюза безопасности: IP-адрес и длина префикса (количество единиц в сетевой маске).

FIXED_2_1_NAME - имя виртуального интерфейса, с которого осуществляется ping для проверки работоспособности основного шлюза безопасности.

FIXED_2_1_PARAM - параметры виртуального интерфейса: IP-адрес и длина префикса (количество единиц в сетевой маске).

CHECK_N - количество неуспешных попыток ping, после которых считается, что связи нет и основной шлюз безопасности вышел из строя. Чем меньше данное число, тем быстрее срабатывает переключение интерфейсов, но тем больше вероятность ложных срабатываний.

Вспомогательный скрипт

Вспомогательный скрипт служит для обеспечения запуска и остановки основного скрипта.

Вспомогательный скрипт сохраним в директории `/etc/init.d` под именем `vpn_reserve_gate_init` и выполним для него

```
gw2#chmod a+x vpn_reserve_gate_init
```

Текст скрипта `vpn_reserve_gate_init`:

```
#!/bin/sh
```

```

# Version 1.2

VPN_RESERVE_GATE_SCRIPT=/etc/init.d/vpn_reserve_gate
PS=/usr/bin/ps
GREP=/usr/bin/grep
KILL=/usr/bin/kill
AWK=/usr/bin/awk

case $1 in
'start')
    $VPN_RESERVE_GATE_SCRIPT &
    ;;
'stop')
    PID=`$PS -ef | $GREP -v grep | $GREP $VPN_RESERVE_GATE_SCRIPT | $AWK '{print $2}'`
    if [ ! -z "$PID" ] ; then
        /usr/bin/kill $PID 1> /dev/null 2>&1
    fi
    ;;
*)
    echo "Usage: vpn_reserve_gate_init { start | stop }"
    ;;
esac

```

Для того, чтобы скрипт `vpn_reserve_gate` вызывался при StartUp-е и останавливался при Shutdown-е, сделаем линки на скрипт `vpn_reserve_gate_init`:

```

gw2#ln -s /etc/init.d/vpn_reserve_gate_init
/etc/rc2.d/S22vpn_reserve_gate_init

gw2#ln -s /etc/init.d/vpn_reserve_gate_init
/etc/rc0.d/K91vpn_reserve_gate_init

```

Создание политики для шлюза безопасности GW2 будем производить аналогично устройству GW1. В Приложении представлен [текст cisco-like конфигурации](#) и [текст LSP](#), [таблица маршрутов](#) и [вывод команды ifconfig -a](#). В выводе зафиксировано состояние интерфейсов для активного и спящего режимов.

Настройка шлюза GW3

Регистрация CA и локального сертификатов производится так же, как и для шлюза безопасности GW1.

[Текст политики безопасности](#), [текст LSP](#), [таблица маршрутов](#) и [вывод команды ifconfig -a](#) для шлюза GW3 приведен в Приложении.

Настройка устройства Router1

В Приложении приведена [таблица маршрутов](#) устройства Router1.

Настройка устройства IPHost1

На устройстве IPHost1 в качестве шлюза по умолчанию нужно указать адрес внутреннего интерфейса шлюза безопасности GW3 - 192.168.103.254.

В [Приложении](#) приведена таблица маршрутизации устройства IPHost1.

Настройка устройства IPHost2

На устройстве IPHost2 в качестве шлюза по умолчанию нужно указать адрес внутреннего интерфейса устройства Router 1 – 10.0.110.254.

В [Приложении](#) приведена таблица маршрутизации устройства IPHost2.

Проверка работоспособности стенда

После того, как настройка GW1, GW2 и GW3 завершена, инициируем создание защищенного соединения.

1. На IPHost2 выполним команду:

```
ping 192.168.103.1
```

```
PING 192.168.103.1 (192.168.103.1): 56 data bytes
64 bytes from 192.168.103.1: icmp_seq=0 ttl=252 time=1851.4 ms
64 bytes from 192.168.103.1: icmp_seq=1 ttl=252 time=859.7 ms
64 bytes from 192.168.103.1: icmp_seq=2 ttl=252 time=1.5 ms
64 bytes from 192.168.103.1: icmp_seq=3 ttl=252 time=1.9 ms
64 bytes from 192.168.103.1: icmp_seq=4 ttl=252 time=3.6 ms
64 bytes from 192.168.103.1: icmp_seq=5 ttl=252 time=3.5 ms
64 bytes from 192.168.103.1: icmp_seq=6 ttl=252 time=2.0 ms
```

В результате выполнения этой команды между устройствами GW1 и GW3 будет установлен VPN туннель. Убедиться в этом можно при помощи утилиты sa_show.

2. На устройстве GW1 выполним команду:

```
Gw1#/opt/VPNagent/bin/sa_show
```

```
IPSec SA Num (Remote Addr,Port)-(Local Addr,Port) Protocol Action
Type Sent Rec
IPSec SA 1 6 (192.168.103.0-192.168.103.255,*)- (10.0.110.0-
10.0.110.255,*) * ESP tunn 588 868
```

Теперь, не останавливая посылку ICMP-пакетов с IPHost2, отключим основное устройство GW1. Дождемся установления SA через резервный шлюз и возобновления связи:

```
64 bytes from 192.168.103.1: icmp_seq=0 ttl=252 time=9.1 ms
64 bytes from 192.168.103.1: icmp_seq=1 ttl=252 time=2.0 ms
64 bytes from 192.168.103.1: icmp_seq=2 ttl=252 time=25.3 ms
64 bytes from 192.168.103.1: icmp_seq=5 ttl=252 time=3.7 ms
64 bytes from 192.168.103.1: icmp_seq=6 ttl=252 time=2.0 ms
```

```
64 bytes from 192.168.103.1: icmp_seq=13 ttl=252 time=5576.7 ms
64 bytes from 192.168.103.1: icmp_seq=14 ttl=252 time=4577.4 ms
64 bytes from 192.168.103.1: icmp_seq=15 ttl=252 time=3577.7 ms
64 bytes from 192.168.103.1: icmp_seq=19 ttl=252 time=2.3 ms
64 bytes from 192.168.103.1: icmp_seq=20 ttl=252 time=5.6 ms
```

В результате выполнения этой команды между устройствами GW2 и GW3 будет установлен VPN туннель. Убедиться в этом можно при помощи утилиты sa_show.

3. На устройстве GW2 выполним команду:

```
Gw2#/opt/VPNagent/bin/sa_show
```

```
IPSec SA Num (Remote Addr,Port)-(Local Addr,Port) Protocol Action
Type Sent Rec
IPSec SA 1 6 (192.168.103.0-192.168.103.255,*)- (10.0.110.0-
10.0.110.255,*) * ESP tunn 327 632
```

После этого вернем стэнд в исходное состояние и наблюдаем переключение с резервного на основной шлюз:

```
64 bytes from 192.168.103.1: icmp_seq=0 ttl=252 time=3.3 ms
64 bytes from 192.168.103.1: icmp_seq=1 ttl=252 time=2.1 ms
64 bytes from 192.168.103.1: icmp_seq=2 ttl=252 time=2.2 ms
64 bytes from 192.168.103.1: icmp_seq=3 ttl=252 time=2.1 ms
```

```

64 bytes from 192.168.103.1: icmp_seq=8 ttl=252 time=4228.4 ms
64 bytes from 192.168.103.1: icmp_seq=9 ttl=252 time=3229.0 ms
64 bytes from 192.168.103.1: icmp_seq=10 ttl=252 time=2229.4 ms
64 bytes from 192.168.103.1: icmp_seq=13 ttl=252 time=20.1 ms
64 bytes from 192.168.103.1: icmp_seq=14 ttl=252 time=2.0 ms

```

Можно убедиться что пакеты идут через основной шлюз. Для этого на GW1 выполним команду:

```
#./sa_show
```

```

IPSec SA Num (Remote Addr,Port)-(Local Addr,Port) Protocol Action
Type Sent Rec
IPSec SA 1 13 (192.168.103.0-192.168.103.255,*)-(10.0.110.0-
10.0.110.255,*) * ESP tunn 588 868

```

Мы только что создали конфигурацию, обеспечивающую взаимодействие между устройствами из подсетей SN1и SN2 по защищенному каналу, с использованием кластера из двух шлюзов.

Приложение

Текст cisco-like конфигурации для GW1

```

crypto ipsec df-bit copy
crypto isakmp identity dn
crypto isakmp keepalive 10 5
username cscons password csp
hostname GW1
enable password csp
!
logging trap debugging
!
crypto isakmp policy 1
  hash md5
  encryption des
  authentication rsa-sig
  group 2
!
crypto ipsec transform-set Gost esp-des
  mode tunnel
!
ip access-list extended SN2toSN1
  permit ip 10.0.110.90 0.0.0.0 192.168.103.1 0.0.0.0
!
crypto map CMAP 1 ipsec-isakmp

```

```

match address SN2toSN1
set transform-set Gost
set pfs group2
set peer 10.12.118.1
!
interface FastEthernet0/0
ip address 10.0.110.101 255.255.255.0
ip address 10.0.110.102 255.255.255.0 secondary
ip address 10.0.110.103 255.255.255.0 secondary
crypto map CMAP
!
crypto ca trustpoint s-terra_technological_trustpoint
crl optional
crypto CA certificate chain s-terra_technological_trustpoint
certificate 551C69033C5A62864FCD2D961858B88B
3082036730820314A00====skip====B9E0C4C06767E9824AAC65993BB3F9F918325EA384EBE
quit

!
ip route 0.0.0.0 0.0.0.0 10.0.110.254 1
end

```

Текст LSP для устройства GW1

```

# This is automatically generated LSP
#
# Conversion Date/Time: Tue Feb 12 18:21:27 2008

GlobalParameters(
  Title = "This LSP was automatically generated by CSP
Converter at Tue Feb 12 18:21:27 2008"
  Version = "2.1"
  CRLHandlingMode = OPTIONAL
  LDAPLogMessageLevel = DEBUG
  SystemLogMessageLevel = DEBUG
  PolicyLogMessageLevel = DEBUG
  CertificatesLogMessageLevel = DEBUG
)

SyslogSettings(
  Server = 127.0.0.1
  Facility = LOG_LOCAL7
)

RoutingTable(

```

```
Routes *=
    Route(
        Destination = 192.168.103.0/24
        Gateway = 10.0.110.254
        Metric = 1
    )
)
IKETransform IKETransform_1
(
    CipherAlg    *= "G2814789CPR01-K256-CBC-65534"
    HashAlg      *= "GR341194CPR01-65534"
    GroupID      *= MODP_1024
    LifetimeSeconds = 86400
)

ESPProposal ESP_Gost
(
    Transform* = ESPTransform
    (
        CipherAlg*      = "G2814789CPR01-K256-CBC-254"
        LifetimeSeconds = 3600
        LifetimeKilobytes = 4608000
    )
)

CertDescription ca
(
    Issuer          *= "1.2.840.113549.1.9.1=support@s-
terra.com,C=RU,ST=Moscow,L=Zelenograd,O=S-Terra CSP,OU=Support Group,CN=Support
Group CA"
    SerialNumber    = "551c69033c5a62864fcd2d961858b88b"
    Subject         *= "1.2.840.113549.1.9.1=support@s-
terra.com,C=RU,ST=Moscow,L=Zelenograd,O=S-Terra CSP,OU=Support Group,CN=Support
Group CA"
)

AuthMethodGOSTSign auth_ca
(
    LocalID        = IdentityEntry( DistinguishedName* = USER_SPECIFIC_DATA )
    DoNotMapRemoteIDToCert = TRUE
    AcceptCredentialFrom    *= ca
    SendRequestMode        = ALWAYS
    SendCertMode           = ALWAYS
)
```

```
IKERule IKE_CMAP_1
(
    Transform* = IKETransform_1
    AggrModeAuthMethod *= auth_ca
    MainModeAuthMethod *= auth_ca
    DoAutopass          = TRUE
    DPDIdleDuration     = 10
    DPDResponseDuration = 5
    DPDRetries          = 5
)

IPsecAction CMAP_1
(
    TunnelingParameters *= TunnelEntry(
        PeerIPAddress = 10.12.118.1

        DFHandling=COPY
    )
    ContainedProposals *= ( ESP_Gost )
    GroupID *= MODP_1024
    IKERule = IKE_CMAP_1
)

FilteringRule Filter_nil_acl_CMAP_1
(
    LocalIPFilter *= FilterEntry( IPAddress *= 10.0.110.0/24 )
    PeerIPFilter  *= FilterEntry( IPAddress *= 192.168.103.0/24 )
    NetworkInterfaces *= "rtls0"
    Action *= ( CMAP_1 )
)

FilteringRule Filter_nil_acl
(
    LocalIPFilter *= FilterEntry( IPAddress *= 0.0.0.0..255.255.255.255 )
    PeerIPFilter  *= FilterEntry( IPAddress *= 0.0.0.0..255.255.255.255 )
    NetworkInterfaces *= "rtls0"
    Action *= ( PASS )
)
```

Текст cisco-like конфигурации для GW2

```
crypto ipsec df-bit copy
crypto isakmp identity dn
username cscons password csp
hostname suppgw2
```

```
enable password csp
!
logging trap debugging
!
crypto isakmp policy 1
  hash md5
  encryption des
  authentication rsa-sig
  group 2
!
crypto ipsec transform-set Gost esp-des
  mode tunnel
!
ip access-list extended SN2toSN1
  permit ip 10.0.110.0 0.0.0.255 192.168.103.0 0.0.0.255
!
crypto map CMAP 1 ipsec-isakmp
  match address SN2toSN1
  set transform-set Gost
  set pfs group2
  set peer 10.12.118.1
!
interface FastEthernet0/0
  ip address 10.0.110.101 255.255.255.0
  ip address 10.0.110.104 255.255.255.0 secondary
  crypto map CMAP
!
crypto ca trustpoint s-terra_technological_trustpoint
  crl optional
crypto CA certificate chain s-terra_technological_trustpoint
certificate 551C69033C5A62864FCD2D961858B88B
3082036730820314A00====skip====824AACC65993BB3F9F918325EA384EBE
quit

!
ip route 0.0.0.0 0.0.0.0 10.0.110.254 1
end
```

Текст LSP для устройства GW2

```
# This is automatically generated LSP
#
# Conversion Date/Time: Mon Feb 11 18:24:31 2008

GlobalParameters(
```

```
Title = "This LSP was automatically generated by CSP
Converter at Mon Feb 11 18:24:31 2008"
Version = "2.1"
CRLHandlingMode = OPTIONAL
LDAPLogMessageLevel = DEBUG
SystemLogMessageLevel = DEBUG
PolicyLogMessageLevel = DEBUG
CertificatesLogMessageLevel = DEBUG
)

SyslogSettings(
  Server = 127.0.0.1
  Facility = LOG_LOCAL7
)

RoutingTable(
  Routes *=
    Route(
      Destination = 0.0.0.0/0
      Gateway = 10.0.110.254
      Metric = 1
    )
)

IKETransform IKETransform_1
(
  CipherAlg *= "G2814789CPR01-K256-CBC-65534"
  HashAlg *= "GR341194CPR01-65534"
  GroupID *= MODP_1024
  LifetimeSeconds = 86400
)

ESPProposal ESP_Gost
(
  Transform* = ESPTransform
  (
    CipherAlg* = "G2814789CPR01-K256-CBC-254"
    LifetimeSeconds = 3600
    LifetimeKilobytes = 4608000
  )
)

CertDescription ca
(
```

```
Issuer          *= "1.2.840.113549.1.9.1=support@s-
terra.com,C=RU,ST=Moscow,L=Zelenograd,O=S-Terra CSP,OU=Support Group,CN=Support
Group CA"
SerialNumber    = "551c69033c5a62864fcd2d961858b88b"
Subject         *= "1.2.840.113549.1.9.1=support@s-
terra.com,C=RU,ST=Moscow,L=Zelenograd,O=S-Terra CSP,OU=Support Group,CN=Support
Group CA"
)
```

```
AuthMethodGOSTSign auth_ca
```

```
(
  LocalID        = IdentityEntry( DistinguishedName* = USER_SPECIFIC_DATA )
  DoNotMapRemoteIDToCert = TRUE
  AcceptCredentialFrom      *= ca
  SendRequestMode          = ALWAYS
  SendCertMode             = ALWAYS
)
```

```
IKERule IKE_CMAP_1
```

```
(
  Transform* = IKETransform_1
  AggrModeAuthMethod *= auth_ca
  MainModeAuthMethod *= auth_ca
  DoAutopass          = TRUE
  DoNotUseDPD         = TRUE
)
```

```
IPsecAction CMAP_1
```

```
(
  TunnelingParameters *= TunnelEntry(
    PeerIPAddress = 10.12.118.1

    DFHandling=COPY
  )
  ContainedProposals *= ( ESP_Gost )
  GroupID *= MODP_1024
  IKERule = IKE_CMAP_1
)
```

```
FilteringRule Filter_nil_acl_CMAP_1
```

```
(
  LocalIPFilter *= FilterEntry( IPAddress *= 10.0.110.0/24 )
  PeerIPFilter  *= FilterEntry( IPAddress *= 192.168.103.0/24 )
  NetworkInterfaces *= "iprb0"
  Action *= ( CMAP_1 )
)
```

```
FilteringRule Filter_nil_acl
(
    LocalIPFilter *= FilterEntry( IPAddress *= 0.0.0.0..255.255.255.255 )
    PeerIPFilter  *= FilterEntry( IPAddress *= 0.0.0.0..255.255.255.255 )
    NetworkInterfaces *= "iprb0"
    Action *= ( PASS )
)
```

Текст cisco-like конфигурации для GW3

```
crypto ipsec df-bit copy
crypto isakmp identity dn
username cscons password csp
hostname GW3
enable password csp
!
logging trap debugging
!
crypto isakmp policy 1
    hash md5
    encryption des
    authentication rsa-sig
    group 2
!
crypto ipsec transform-set GOST esp-des
    mode tunnel
!
ip access-list extended cacl
    permit ip 192.168.103.0 0.0.0.255 10.0.110.0 0.0.0.255
!
crypto map CMAP 1 ipsec-isakmp
    match address cacl
    set transform-set GOST
    set pfs group2
    set peer 10.0.110.101
!
interface FastEthernet0/0
    ip address 10.12.118.1 255.255.255.0
    crypto map CMAP
interface FastEthernet0/1
    ip address 192.168.103.254 255.255.255.0
!
crypto ca trustpoint s-terra_technological_trustpoint
    crl optional
```

```
crypto CA certificate chain s-terra_technological_trustpoint
certificate 551C69033C5A62864FCD2D961858B88B
3082036730820314A0====skip====EB9E0C4C06767E9824AAC65993BB3F9F918325EA384EBE
quit

!
ip route 0.0.0.0 0.0.0.0 10.12.118.254 1
end
```

Текст LSP для устройства GW3

```
# This is automatically generated LSP
#
# Conversion Date/Time:   Wed Feb 13 15:56:41 2008

GlobalParameters(
    Title = "This LSP was automatically generated by CSP
Converter at Wed Feb 13 15:56:41 2008"
    Version = "2.1"
    CRLHandlingMode = OPTIONAL
    LDAPLogMessageLevel = DEBUG
    SystemLogMessageLevel = DEBUG
    PolicyLogMessageLevel = DEBUG
    CertificatesLogMessageLevel = DEBUG
)

SyslogSettings(
    Server = 127.0.0.1
    Facility = LOG_LOCAL7
)

RoutingTable(
    Routes *=
        Route(
            Destination = 0.0.0.0/0
            Gateway = 10.12.118.254
            Metric = 1
        )
)

IKETransform IKETransform_1
(
    CipherAlg *= "G2814789CPR01-K256-CBC-65534"
    HashAlg *= "GR341194CPR01-65534"
    GroupID *= MODP_1024
    LifetimeSeconds = 86400
)
```

```
)

ESPProposal ESP_GOST
(
    Transform* = ESPTransform
    (
        CipherAlg* = "G2814789CPR01-K256-CBC-254"
        LifetimeSeconds = 3600
        LifetimeKilobytes = 4608000
    )
)

CertDescription ca
(
    Issuer *= "1.2.840.113549.1.9.1=support@s-
terra.com,C=RU,ST=Moscow,L=Zelenograd,O=S-Terra CSP,OU=Support Group,CN=Support
Group CA"
    SerialNumber = "551c69033c5a62864fcd2d961858b88b"
    Subject *= "1.2.840.113549.1.9.1=support@s-
terra.com,C=RU,ST=Moscow,L=Zelenograd,O=S-Terra CSP,OU=Support Group,CN=Support
Group CA"
)

AuthMethodGOSTSign auth_ca
(
    LocalID = IdentityEntry( DistinguishedName* = USER_SPECIFIC_DATA )
    DoNotMapRemoteIDToCert = TRUE
    AcceptCredentialFrom *= ca
    SendRequestMode = ALWAYS
    SendCertMode = ALWAYS
)

IKERule IKE_CMAP_1
(
    Transform* = IKETransform_1
    AggrModeAuthMethod *= auth_ca
    MainModeAuthMethod *= auth_ca
    DoAutopass = TRUE
    DoNotUseDPD = TRUE
)

IPsecAction CMAP_1
(
    TunnelingParameters *= TunnelEntry(
        PeerIPAddress = 10.0.110.101
```

```

        DFHandling=COPY
    )
    ContainedProposals *= ( ESP_GOST )
    GroupID *= MODP_1024
    IKERule = IKE_CMAP_1
)

FilteringRule Filter_nil_acl_CMAP_1
(
    LocalIPFilter *= FilterEntry( IPAddress *= 192.168.103.0/24 )
    PeerIPFilter  *= FilterEntry( IPAddress *= 10.0.110.0/24 )
    NetworkInterfaces *= "rtls0"
    Action *= ( CMAP_1 )
)

FilteringRule Filter_nil_acl
(
    LocalIPFilter *= FilterEntry( IPAddress *= 0.0.0.0..255.255.255.255 )
    PeerIPFilter  *= FilterEntry( IPAddress *= 0.0.0.0..255.255.255.255 )
    NetworkInterfaces *= "rtls0"
    Action *= ( PASS )
)

FilteringRule Filter_nil_acl_1
(
    LocalIPFilter *= FilterEntry( IPAddress *= 0.0.0.0..255.255.255.255 )
    PeerIPFilter  *= FilterEntry( IPAddress *= 0.0.0.0..255.255.255.255 )
    NetworkInterfaces *= "rtls1"
    Action *= ( PASS )
)

```

Таблица маршрутизации GW1

```
gwl#netstat -rn
```

```
Routing Table: IPv4
```

Destination	Gateway	Flags	Ref	Use	Interface
10.0.177.1	10.0.110.177	UGH	1	79	
10.0.110.0	10.0.110.101	U	1	11	rtls0
10.0.110.0	10.0.110.101	U	1	0	rtls0:1
10.0.110.0	10.0.110.101	U	1	0	rtls0:2
default	10.0.110.254	UG	1	4	
127.0.0.1	127.0.0.1	UH	2	1292	lo0

Таблица маршрутизации GW2

```
gw2#netstat -rn
```

```
Routing Table: IPv4
```

Destination	Gateway	Flags	Ref	Use	Interface
10.0.177.1	10.0.110.177	UGH	1	42	
10.0.110.0	10.0.110.104	U	1	8	iprb0:1
192.168.2.0	172.16.1.2	UG	1	248434	
default	10.0.110.254	UG	1	2	
127.0.0.1	127.0.0.1	UH	2	422	lo0

Таблица маршрутизации GW3

```
gw3#netstat -rn
```

```
Routing Table: IPv4
```

Destination	Gateway	Flags	Ref	Use	Interface
192.168.103.0	192.168.103.254	U	1	28	rtls1
10.12.118.0	10.12.118.1	U	1	39	rtls0
224.0.0.0	10.12.118.1	U	1	0	rtls0
default	10.12.118.254	UG	1	25	
127.0.0.1	127.0.0.1	UH	2	445	lo0

Таблица маршрутизации Router1

```
router1#netstat -rn
```

```
Kernel IP routing table
```

Destination	Gateway	Genmask	Flags	MSS Window	irtt	Iface
10.12.118.0	0.0.0.0	255.255.255.0	U	0 0	0	eth1
192.168.103.0	10.12.118.1	255.255.255.0	UG	0 0	0	eth1
10.0.110.0	0.0.0.0	255.255.255.0	U	0 0	0	eth0
0.0.0.0	10.0.110.101	0.0.0.0	UG	0 0	0	eth0

Вывод команды ifconfig –а устройства GW1

```
gw1#ifconfig -a
```

```
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
    inet 127.0.0.1 netmask ff000000
rtls0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
    inet 10.0.110.101 netmask ffffffff broadcast 10.0.110.255
    ether 0:50:fc:90:7f:1c
```

```

rtls0:1: flags=1040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4> mtu 1500
index 2
    inet 10.0.110.102 netmask ffffffff broadcast 10.0.110.255
rtls0:2: flags=1040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4> mtu 1500
index 2
    inet 10.0.110.103 netmask ffffffff broadcast 10.0.110.255

```

Вывод команды ifconfig –а устройства GW2

Спящий режим:

```

gw2#ifconfig -a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
    inet 127.0.0.1 netmask ff000000
iprb0: flags=1000842<BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
    inet 10.0.110.101 netmask ffffffff broadcast 10.0.110.255
    ether 0:a0:c9:85:6f:17
iprb0:1: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
    inet 10.0.110.104 netmask ffffffff broadcast 10.0.110.255

```

Активный режим:

```

gw2#ifconfig -a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
    inet 127.0.0.1 netmask ff000000
iprb0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
    inet 10.0.110.101 netmask ffffffff broadcast 10.0.110.255
    ether 0:a0:c9:85:6f:17
iprb0:1: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
    inet 10.0.110.104 netmask ffffffff broadcast 10.0.110.255

```

Вывод команды ifconfig –а устройства GW3

```

gw3#ifconfig -a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
    inet 127.0.0.1 netmask ff000000
rtls0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
    inet 10.12.118.1 netmask ffffffff broadcast 10.12.118.255
    ether 0:30:18:4a:31:f8
rtls1: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 3
    inet 192.168.103.254 netmask ffffffff broadcast 192.168.103.255
    ether 0:30:18:4a:31:f7

```

Таблица маршрутизации IPHost1

```

iphost1# netstat -rn
Kernel IP routing table

```

Destination	Gateway	Genmask	Flags	MSS Window	irtt	Iface
192.168.103.0	0.0.0.0	255.255.255.0	U	0 0	0	eth0
0.0.0.0	192.168.103.254	0.0.0.0	UG	0 0	0	eth0

Таблица маршрутизации IPHost2

```
iphost1#netstat -rn
Kernel IP routing table
Destination      Gateway          Genmask         Flags       MSS Window  irtt Iface
192.168.103.0    10.0.110.101    255.255.255.0  UG          0 0         0 eth0
10.0.110.0       0.0.0.0         255.255.255.0  U           0 0         0 eth0
0.0.0.0          10.0.110.254    0.0.0.0        UG          0 0         0 eth0
```