

Отказоустойчивое решение (кластер). Аутентификация на сертификатах

Описание стенда

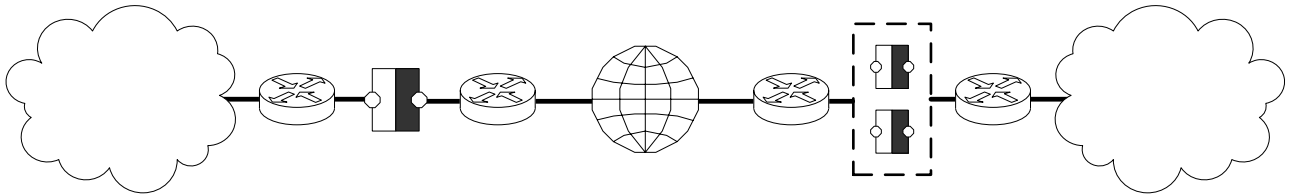


Рис. 1. Общая схема подключения

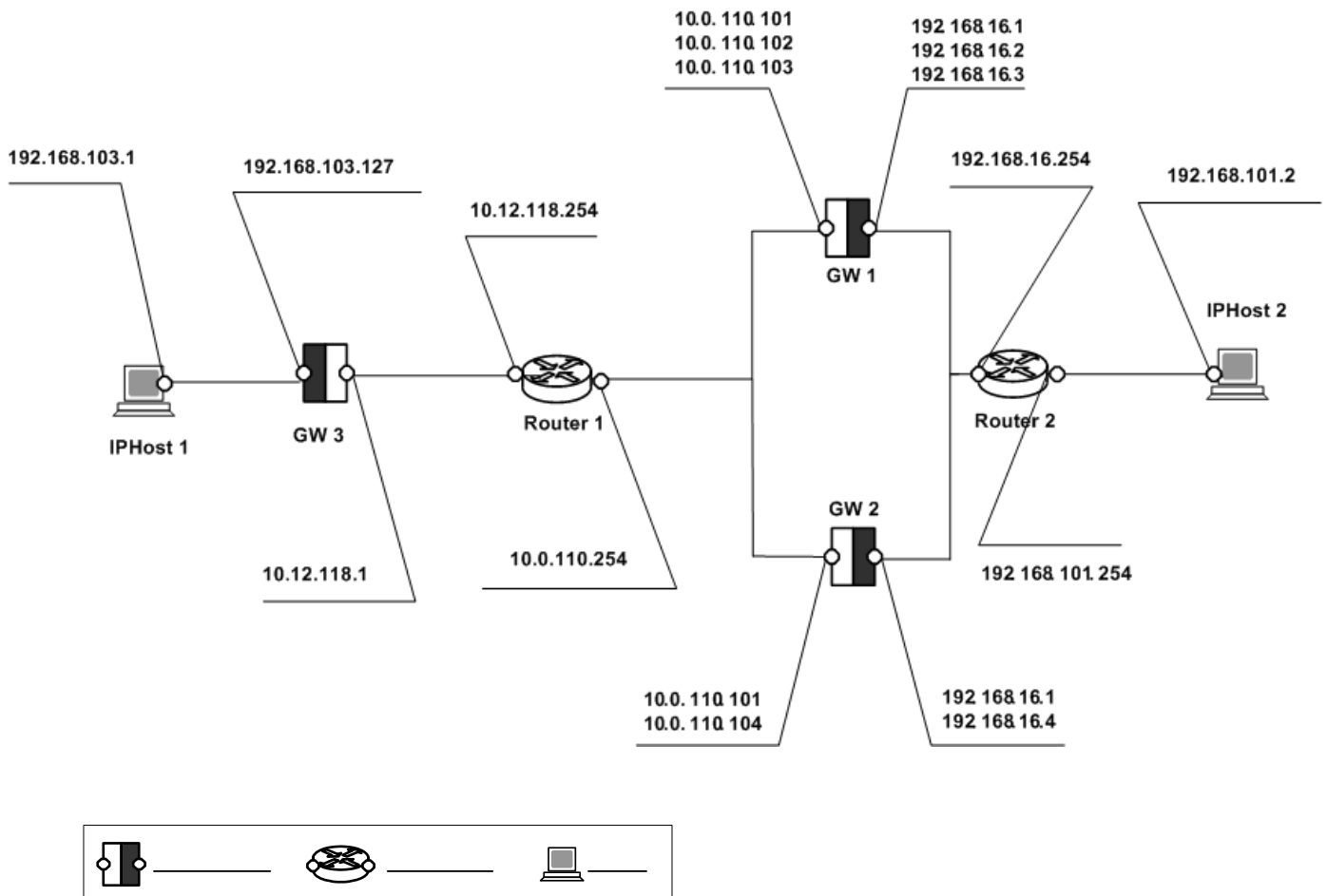


Рис. 2 Схема стенда

Сценарий представляет собой две подсети, одна из которых SN1-192.168.103.0/24 защищается шлюзом безопасности GW3, а вторая подсеть SN2-192.168.101.0/24 защищается кластером, функции которого выполняют два шлюза безопасности GW1 и GW2. На шлюзах установлен продукт CSP VPN Gate. В схему включены два роутера: Router1 и Router2.

Логика работы отказоустойчивого решения

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 определены два «надежных» устройства. На эти устройства будут отправляться ICMP-пакеты для диагностики работоспособности сетевых интерфейсов шлюза безопасности.
- В нормальном режиме весь трафик обрабатывается на основном шлюзе безопасности. Резервный шлюз безопасности в это время находится в режиме ожидания.
- В случае выхода из строя основного шлюза безопасности его заменяет резервный шлюз безопасности и обрабатывает весь проходящий трафик.
- После восстановления работоспособности основного шлюза безопасности резервный шлюз переходит в режим ожидания и отдает управление основному шлюзу безопасности.
- Проверка работоспособности основного и резервного шлюзов безопасности, а также действия по активации и настройке интерфейсов производятся с помощью скриптов, устанавливаемых на шлюзы безопасности.

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

Основные шаги по настройке устройств стенда

Настройка устройств стенда:

- Настройка шлюзов безопасности GW1 – GW3:
 - Настройка адресов шлюза безопасности
 - Установка скриптов
 - Настройка роутинга. Установка роутера по умолчанию
 - Создание политики безопасности, используя командную строку
 - Регистрация CA и локального сертификатов
- Настройка Router1 - Router2
- Настройка IPHost1 - IPHost2.

Настройка 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` (в ОС Linux - `/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`, а интерфейсу с именем `PHYS_1_2_NAME` – адрес `FIXED_1_2_PARAM`. Соответствие имен, адресов и конкретных значений смотрите в разделе [“Описание настраиваемых параметров основного скрипта устройства GW1”](#).

Установка скриптов

На шлюз безопасности GW1 нужно установить два скрипта. Один скрипт будет отвечать за конфигурирование интерфейсов и оценку работоспособности шлюза, а второй скрипт будет отвечать за старт основного скрипта при перезагрузке операционной системы.

Основной скрипт

Основной скрипт сохраняем в директории `/etc/init.d` под именем `vpn_main_gate` и выполняем для него

```
chmod a+x vpn_main_gate
```

Текст скрипта `vpn_main_gate`:

```
#!/bin/ksh

# Version 1.2

IP_RELIABLE1=10.0.110.254
IP_RELIABLE2=192.168.16.254

PHYS_1_1_NAME=iprb0
TUNNEL_1_PARAM=10.0.110.101/16

TO_CHECK_1_NAME=$PHYS_1_1_NAME:1
TO_CHECK_1_PARAM=10.0.110.102/16

FIXED_1_1_NAME=$PHYS_1_1_NAME:2
FIXED_1_1_PARAM=10.0.110.103/16

PHYS_1_2_NAME=iprb1

TUNNEL_2_PARAM=192.168.16.1/24

TO_CHECK_2_NAME=$PHYS_1_2_NAME:1
```

```
TO_CHECK_2_PARAM=192.168.16.2/24

FIXED_1_2_NAME=$PHYS_1_2_NAME:2
FIXED_1_2_PARAM=192.168.16.3/24

CHECK_N=3
RELAX_TIMEOUT=4

PING=/usr/sbin/ping

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

do_ping2()
{
    $PING $IP_RELIABLE2 1 > /dev/null
}

notify()
{
    logger -p local0.notice "$1"
    echo `date` "$1" > /dev/console
}

on_fail()
{
    ifconfig $PHYS_1_1_NAME down 2> /dev/null
    ifconfig $PHYS_1_2_NAME down 2> /dev/null
    ifconfig $TO_CHECK_1_NAME down 2> /dev/null
    ifconfig $TO_CHECK_2_NAME down 2> /dev/null
    ifconfig $FIXED_1_1_NAME $FIXED_1_1_PARAM broadcast + -deprecated 2>
/dev/null
    ifconfig $FIXED_1_2_NAME $FIXED_1_2_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
    ifconfig $TO_CHECK_2_NAME $TO_CHECK_2_PARAM broadcast + deprecated up 2>
/dev/null
}
```

```

sleep $RELAX_TIMEOUT
ifconfig $PHYS_1_1_NAME $TUNNEL_1_PARAM broadcast + up 2> /dev/null
ifconfig $PHYS_1_2_NAME $TUNNEL_2_PARAM broadcast + up 2> /dev/null
ifconfig $FIXED_1_1_NAME deprecated 2> /dev/null
ifconfig $FIXED_1_2_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 $TO_CHECK_2_NAME plumb 2> /dev/null

ifconfig $FIXED_1_1_NAME plumb 2> /dev/null
ifconfig $FIXED_1_2_NAME plumb 2> /dev/null

ifconfig $PHYS_1_1_NAME $TUNNEL_1_PARAM down broadcast + 2> /dev/null
ifconfig $PHYS_1_2_NAME $TUNNEL_2_PARAM down broadcast + 2> /dev/null

ifconfig $FIXED_1_1_NAME $FIXED_1_1_PARAM up broadcast + 2> /dev/null
ifconfig $FIXED_1_2_NAME $FIXED_1_2_PARAM up broadcast + 2> /dev/null

if do_ping1 && do_ping2; 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_ping2; 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 && do_ping2; then
        RETRY_COUNT1=$CHECK_N
        RETRY_COUNT2=$CHECK_N
        on_ok
    fi
fi
done

```

Описание настраиваемых параметров основного скрипта устройства GW1

В основном скрипте настройте следующие параметры:

IP_RELIABLE1 – IP-адрес надежного устройства из внешней подсети.

IP_RELIABLE2 – IP-адрес надежного устройства из внутренней подсети.

PHYS_1_1_NAME – имя внешнего физического интерфейса.

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

TO_CHECK_1_NAME – имя внешнего 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 внешнего виртуального интерфейса: IP-адрес и длина префикса (количество единиц в сетевой маске).

FIXED_1_1_NAME – имя внешнего виртуального интерфейса Virtual (fixed).

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

PHYS_1_2_NAME – имя внутреннего физического интерфейса.

TUNNEL_2_PARAM – параметры внутреннего физического интерфейса: IP-адрес и длина префикса (количество единиц в сетевой маске).

TO_CHECK_2_NAME – имя внутреннего виртуального интерфейса Virtual (to check), который будет использоваться для определения работоспособности шлюза безопасности (на него будет отправлять ping резервный шлюз безопасности).

TO_CHECK_2_PARAM – параметры внутреннего виртуального интерфейса: IP-адрес и длина префикса (количество единиц в сетевой маске).

FIXED_1_2_NAME – имя внутреннего виртуального интерфейса Virtual (fixed).

FIXED_1_2_PARAM – параметры внутреннего виртуального интерфейса Virtual (fixed): IP-адрес и длина префикса.

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 туннельных интерфейсов iprb0 и iprb1. В этот таймаут резервный шлюз безопасности должен опустить туннельные интерфейсы iprb0 и iprb1.

notify – команды оповещения. Настройка описана ниже в разделе “Команды оповещения”.

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, для чего следует выполнить:

```
/etc/init.d/vpn_main_gate_init start | stop
```

Команды оповещения

Еще одной настройкой основного скрипта является функция оповещения (notify).

Она позволяет задавать команды оповещения такие как вывод в системную консоль, syslog и т.п.

В основных скриптах эти команды пишутся в блоке:

```
notify()
{
# Команды оповещения.
}
```

В данную функцию передается строка сообщения для вывода. В командах эта строка задается как \$1.

Пример команды вывода на системную консоль. Вместе с сообщением выводится дата и время события:

```
echo `date` "$1" > /dev/console
```

Пример команды вывода в syslog:

```
logger -p local0.notice "$1"
```

Примечание: в аргументе –p задается syslog facility и severity генерируемого сообщения. Чтобы данное сообщение было получено, необходимо, чтобы syslog-daemon был соответствующим образом сконфигурирован в файле /etc/syslog.conf.

Пример вывода сообщений с facility local0 и severity notice в файл /var/adm/messages:

```
local0.notice /var/adm/vpnlog
```

Пример отправки сообщений на хост 192.168.101.2:

```
local0.notice @192.168.101.2
```

Пример функции вывода на системную консоль и в syslog:

```
notify()
{
logger -p local0.notice "$1"
echo `date` "$1" > /dev/console
}
```

Вспомогательный скрипт

Вспомогательный скрипт служит для обеспечения запуска и остановки основного скрипта.

Вспомогательный скрипт сохраняем в директории /etc/init.d под именем vpn_main_gate_init и выполняем для него

```
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, необходимо сделать линки на скрипт `vpn_main_gate_init`:

```
ln -s /etc/init.d/vpn_main_gate_init /etc/rc2.d/S22vpn_main_gate_init
ln -s /etc/init.d/vpn_main_gate_init /etc/rc0.d/K91vpn_main_gate_init
```

Установка шлюза по умолчанию

После выполнения этих настроек необходимо установить шлюз по умолчанию. В качестве этого устройства следует выбрать адрес устройства Router1: 10.0.110.254.

Создание политики GW1

Приступим к настройке CSP VPN Gate, установленного на шлюзе безопасности GW1.

Конфигурирование можно производить локально или удаленно, используя командную строку. Для этого перейдем в директорию `/opt/VPNagent/bin/` и запустим `cs_console`.

При помощи этого приложения необходимо набрать следующую конфигурацию:

```
crypto ipsec df-bit clear
crypto isakmp identity dn
```

```
crypto isakmp keepalive 10 3
username cscons password csp
hostname GW1
enable password csp
!
!
!
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 CryptoACL
  permit ip 192.168.101.0 0.0.0.255 192.168.103.0 0.0.0.255
!
crypto map CMAP 1 ipsec-isakmp
  match address CryptoACL
  set transform-set GOST
  set pfs group2
  set peer 10.12.118.1
!
!
!
interface FastEthernet0/0
  ip address 10.0.110.101 255.255.0.0
  ip address 10.0.110.102 255.255.0.0 secondary
  ip address 10.0.110.103 255.255.0.0 secondary
  crypto map CMAP
!
interface FastEthernet0/1
  ip address 192.168.16.1 255.255.255.0
  ip address 192.168.16.2 255.255.255.0 secondary
  ip address 192.168.16.3 255.255.255.0 secondary
```

(*Следующий блок команд производит регистрацию CA сертификата, перед набором этих команд прочитайте раздел «Регистрация CA сертификата»*).

```
crypto ca trustpoint s_terraCSP
  crl optional
crypto CA certificate chain s_terraCSP
certificate 001
```

```
[CA_CERTIFICATE_HEX_STRING]
```

Quit

Текст LSP конфигурации, которая получена в результате конвертирования cisco-like конфигурации, приведена в Приложении. Ее текст можно получить, используя утилиту `lsp_mgr show`, входящую в состав CSP VPN Gate.

Регистрация CA сертификата

Перед началом регистрации CA сертификата в продукте на шлюзе безопасности GW1 надо скопировать CA сертификат в файл и преобразовать его из бинарного представления в шестнадцатеричное (формат DER encoded binary X.509 (.CER)). Воспользуйтесь, например, утилитой `bin2hex.exe` (`ca.cer` – имя файла с CA сертификатом):

```
bin2hex ca.cer
```

Получим примерно следующее:

```
308202B530820257A003020102021002014FFEFDA2AD964CFC68B89AA59EC8300E060A2B0601
0401AD590103020...
```

Нам нужно, чтобы шестнадцатеричное представление сертификата было в виде одной строки, для этого можно сделать `copy/past` в Wordpad и там удалить символы перевода строк. Полученную HEX-строку можно сохранить в файл.

В блоке команд регистрации CA сертификата вместо `[CA_CERTIFICATE_HEX_STRING]` нужно вставить полученное шестнадцатеричное представление CA сертификата в виде одной строки.

Создание локального сертификата

Создание локального сертификата мы здесь не описываем. Подробное описание процедуры генерации ключевой пары, формирования запроса на сертификат и создания локального сертификата приведено в документе `CSP_VPN_Appendix`.

Регистрация локального сертификата

Доставьте на шлюз безопасности GW1 контейнер и локальный сертификат для этого шлюза безопасности, скопированный в файл.

Если используется СКЗИ «Крипто-КОМ 3.1», то контейнер с ключевой информацией представляет собой каталог. В этом случае для доставки локального сертификата `gate1.cer` и контейнера `gate1` на шлюз можно воспользоваться утилитой `pscp.exe` из пакета Putty:



Среда передачи в этом случае должна быть доверенной

Предупреждение

Для доставки, например, в каталог `/certif` выполните:

```
pscp -r gate1.cer root@10.0.110.101:/certif
```

```
pscp -r gate1 root@10.0.110.101:/certif
```

Регистрация локального сертификата на шлюзе производится утилитой `cert_mgr import`, входящей в состав продукта CSP VPN Gate:

```
/opt/VPNagent/bin/cert_mgr import -f /certif/gate1.cer -kc /certif/gate1
```

Если используется СКЗИ «КриптоПро CSP» и контейнер с секретным ключом расположен на диске, а на аппаратной платформе присутствует флоппи-дискковод, то локальный сертификат регистрируется командой:

```
/opt/VPNagent/bin/cert_mgr import -f /certif/gatel.cer -kc  
'FAT12\\gatel.000'
```

Настройка шлюза безопасности GW2

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

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` (в ОС Linux - `/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`, а интерфейсу с именем `PHYS_2_2_NAME` – адрес `FIXED_2_2_PARAM`. Соответствие имен, адресов и конкретных значений смотрите в разделе ["Описание настраиваемых параметров основного скрипта устройства GW2"](#).

Основной скрипт

Основной скрипт надо положить в директорию `/etc/init.d` под именем `vpn_reserve_gate` и выполнить для него

```
chmod a+x vpn_reserve_gate
```

Текст скрипта `vpn_reserve_gate`:

```
#!/bin/ksh  
# Version 1.2  
  
IP_RELIABLE1=10.0.110.254  
IP_RELIABLE2=192.168.16.254  
  
IP_TO_CHECK1=10.0.110.102  
IP_TO_CHECK2=192.168.16.1  
  
PHYS_2_1_NAME=iprb0  
  
TUNNEL_1_PARAM=10.0.110.101/16  
  
FIXED_2_1_NAME=$PHYS_2_1_NAME:1  
FIXED_2_1_PARAM=10.0.110.104/16  
  
PHYS_2_2_NAME=iprb1  
  
TUNNEL_2_PARAM=192.168.16.1/24
```

```
FIXED_2_2_NAME=$PHYS_2_2_NAME:1
FIXED_2_2_PARAM=192.168.16.4/24

CHECK_N=3

PING=/usr/sbin/ping

notify()
{
    logger -p local0.notice "$1"
    echo `date` "$1" > /dev/console
}

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

do_ping2()
{
    $PING $IP_RELIABLE2 1 > /dev/null
}

do_ping_g1()
{
    $PING $IP_TO_CHECK1 1 > /dev/null
}

do_ping_g2()
{
    $PING $IP_TO_CHECK2 1 > /dev/null
}

ifs_down()
{
    ifconfig $FIXED_2_1_NAME -deprecated 2> /dev/null
    ifconfig $FIXED_2_2_NAME -deprecated 2> /dev/null
    ifconfig $PHYS_2_1_NAME down 2> /dev/null
    ifconfig $PHYS_2_2_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 $PHYS_2_2_NAME $TUNNEL_2_PARAM broadcast + up 2> /dev/null
    ifconfig $FIXED_2_1_NAME deprecated 2> /dev/null
    ifconfig $FIXED_2_2_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 $FIXED_2_2_NAME plumb 2> /dev/null
ifconfig $FIXED_2_2_NAME $FIXED_2_2_PARAM up broadcast + 2> /dev/null

ifconfig $PHYS_2_1_NAME $TUNNEL_1_PARAM down broadcast + 2> /dev/null
ifconfig $PHYS_2_2_NAME $TUNNEL_2_PARAM down broadcast + 2> /dev/null

if do_ping_g1 && do_ping_g2; 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 && do_ping2; 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 do_ping_g2; then
        RETRY_COUNT_G2=$CHECK_N
        RETRY_COUNT2=$CHECK_N
    else
        RETRY_COUNT_G2=$((RETRY_COUNT_G2-1))
    fi

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

else
    if do_ping_g1 && do_ping_g2; 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

    if do_ping2; then
        RETRY_COUNT2=$CHECK_N
    else

```

```
        RETRY_COUNT2=$((RETRY_COUNT2-1))
        if [[ $RETRY_COUNT1 -gt 0 ]] && [[ $RETRY_COUNT2 -eq 0 ]];
then
            ifs_down
            IFS_UP_FLAG=0
            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 && do_ping2 && [[ $IFS_UP_FLAG -eq 0 ]]; then
            RETRY_COUNT1=$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 из внешней подсети.

IP_RELIABLE2 – IP адрес надежного устройства из внутренней подсети.

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

IP_TO_CHECK2 – адрес внутреннего интерфейса основного шлюза безопасности, по которому производится проверка его работоспособности.

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

PHYS_2_2_NAME – имя внутреннего физического интерфейса резервного шлюза безопасности, на котором установлен туннельный адрес (общий для обоих шлюзов безопасности).

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

TUNNEL_2_PARAM – параметры внутреннего физического интерфейса резервного шлюза безопасности: IP-адрес и длина префикса (количество единиц в сетевой маске).

FIXED_2_1_NAME – имя внешнего виртуального интерфейса, с которого осуществляется ring для проверки работоспособности основного шлюза безопасности.

FIXED_2_2_NAME – имя внутреннего виртуального интерфейса, с которого осуществляется ring для проверки работоспособности основного шлюза безопасности.

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

FIXED_2_2_PARAM – параметры внутреннего виртуального интерфейса: IP-адрес и длина префикса (количество единиц в сетевой маске).

CHECK_N – количество неуспешных попыток ring, 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.

notify – команды оповещения.

Команды оповещения

Настройка команд оповещения производится точно так же, как и для основного скрипта шлюза безопасности GW1.

Вспомогательный скрипт

Вспомогательный скрипт служит для обеспечения запуска и остановки основного скрипта.

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

```
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 необходимо сделать линки на скрипт `vpn_reserve_gate_init`:

```
ln -s /etc/init.d/vpn_reserve_gate_init /etc/rc2.d/S22vpn_reserve_gate_init
ln -s /etc/init.d/vpn_reserve_gate_init /etc/rc0.d/K91vpn_reserve_gate_init
```

Установка шлюза по умолчанию

После выполнения этих настроек необходимо установить шлюз по умолчанию. В качестве этого устройства следует выбрать адрес устройства `Router1` - `10.0.110.254`.

Создание политики GW2

Настройки CSP VPN Gate устройства `GW2` ни чем не отличаются от настроек устройства `GW1` и выполняются аналогично. Конфигурация этого устройства приведена в приложении.

```
crypto ipsec df-bit clear
crypto isakmp identity dn
crypto isakmp keepalive 10 3
username cscons password csp
hostname GW2
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 CryptoACL
  permit ip 192.168.101.0 0.0.0.255 192.168.103.0 0.0.0.255
!
crypto map CMAP 1 ipsec-isakmp
  match address CryptoACL
  set transform-set GOST
```

```
set pfs group2
set peer 10.12.118.1
!
!
!
interface FastEthernet0/0
ip address 10.0.110.101 255.255.0.0
ip address 10.0.110.104 255.255.0.0 secondary
crypto map CMAP
interface FastEthernet0/1
ip address 192.168.16.1 255.255.255.0
ip address 192.168.16.4 255.255.255.0 secondary
!
crypto ca trustpoint s_terraCSP
crl optional
crypto CA certificate chain s_terraCSP
certificate 001
[CA_CERTIFICATE_HEX_STRING]
quit
```

Текст LSP конфигурации, которая получена в результате конвертирования cisco-like конфигурации, приведена в Приложении. Ее текст можно получить, используя утилиту `lsp_mgr show`, входящую в состав CSP VPN Gate.

Регистрация CA сертификата

Подготовка CA сертификата к регистрации в продукте производится также как и на шлюзе безопасности GW1.

Регистрация локального сертификата

Доставка локального сертификата и контейнера на шлюз безопасности GW2, а также регистрация сертификата производится аналогично, описанному для шлюза GW1.

Настройка шлюза безопасности GW3

Установка шлюза по умолчанию

На это устройство скрипты не устанавливаются, нужно только указать шлюз по умолчанию - 10.12.118.254.

Создание политики GW3

Для этого необходимо перейти в директорию `/opt/VPNagent/bin` и запустить `cs_console`.

В консоли нужно набрать конфигурацию:

```
crypto ipsec df-bit clear
crypto isakmp identity dn
crypto isakmp keepalive 10 3
```

```
username cicons 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 CryptoACL
  permit ip 192.168.103.0 0.0.0.255 192.168.101.0 0.0.0.255
!
crypto map CMAP 1 ipsec-isakmp
  match address CryptoACL
  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 192.168.103.127 255.255.255.0
!
crypto ca trustpoint s_terraCSP
  crl optional
crypto CA certificate chain s_terraCSP
certificate 001
[CA_CERTIFICATE_HEX_STRING]
quit
```

Текст LSP конфигурации, которая получена в результате конвертирования cisco-like конфигурации, приведена в Приложении. Ее текст можно получить, используя утилиту `lsp_mgr show`, входящую в состав CSP VPN Gate.

Регистрация CA сертификата

Подготовка CA сертификата к регистрации в продукте производится также как и на шлюзе безопасности GW1.

Регистрация локального сертификата

Доставка локального сертификата и контейнера на шлюз безопасности GW3, а также регистрация сертификата производятся аналогично, описанному для шлюза GW1.

Настройка устройства Router1

Дополнительных настроек для этого устройства не требуется.

Настройка устройства Router2

Настройка этого устройства сводится к установке в качестве шлюза по умолчанию «общего» адреса устройств GW1 и GW2 в этом сегменте сети - 192.168.16.1.

Настройка устройства IPHost1

Настройка этого устройства сводится к установке в качестве шлюза по умолчанию «внутреннего» адреса устройства GW3: 192.168.103.127.

Настройка устройства IPHost2

Настройка этого устройства сводится к установке в качестве шлюза по умолчанию «внутреннего» адреса устройства Router2: 192.168.101.254.

Приложение

LSP устройства GW1

```
# This is automatically generated LSP
# Conversion Date/Time: Mon Apr 24 18:04:48 2006

GlobalParameters(
  Title = "This LSP was automatically generated by
CSP Converter at Mon Apr 24 18:04:48 2006"
  Version = "2.1"
  CRLHandlingMode = OPTIONAL
  LDAPLogMessageLevel = INFO
  SystemLogMessageLevel = INFO
  PolicyLogMessageLevel = INFO
  CertificatesLogMessageLevel = INFO
)
SyslogSettings(
  Server = 127.0.0.1
  Facility = LOG_LOCAL7
)
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      *= "C=RU,O=GINS,OU=QA,CN=MSCA"
  SerialNumber = "33ae6c6b85536f834a8d8e5358333f4f"
  Subject     *= "C=RU,O=GINS,OU=QA,CN=MSCA"
)
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
  DPDIIdleDuration   = 10
  DPDResponseDuration = 3
  DPDRetries         = 5
)
IPsecAction CMAP_1(
  TunnelingParameters *= TunnelEntry(
    PeerIPAddress = 10.12.118.1
    DFHandling=CLEAR
  )
  ContainedProposals *= ( ESP_GOST )
  GroupID *= MODP_1024
  IKERule = IKE_CMAP_1
)
FilteringRule Filter_nil_acl_CMAP_1(
  LocalIPFilter *= FilterEntry( IPAddress *= 192.168.101.0/24 )
  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 )
)

```

LSP устройства GW2

```

# This is automatically generated LSP
# Conversion Date/Time:  Mon Apr 24 17:15:35 2006

GlobalParameters(
    Title = "This LSP was automatically generated by
CSP Converter at Mon Apr 24 17:15:35 2006"
    Version = "2.1"
    CRLHandlingMode = OPTIONAL
    LDAPLogMessageLevel = DEBUG
    SystemLogMessageLevel = DEBUG
    PolicyLogMessageLevel = DEBUG
    CertificatesLogMessageLevel = DEBUG
)
SyslogSettings(
    Server = 127.0.0.1
    Facility = LOG_LOCAL7
)
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      *= "C=RU,O=GINS,OU=QA,CN=MSCA"
    SerialNumber = "33ae6c6b85536f834a8d8e5358333f4f"
    Subject     *= "C=RU,O=GINS,OU=QA,CN=MSCA"
)
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
    DPDIIdleDuration  = 10
    DPDResponseDuration = 3
    DPDRetries        = 5
)
IPsecAction CMAP_1(
    TunnelingParameters *= TunnelEntry(
        PeerIPAddress = 10.12.118.1
        DFHandling=CLEAR
    )
    ContainedProposals *= ( ESP_GOST )
    GroupID *= MODP_1024
    IKERule = IKE_CMAP_1
)
FilteringRule Filter_nil_acl_CMAP_1(
    LocalIPFilter *= FilterEntry( IPAddress *= 192.168.101.0/24 )
    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 )
)
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 *= "iprb1"
    Action *= ( PASS )
)
```

```
# This is automatically generated LSP
# Conversion Date/Time: Mon Apr 24 17:04:54 2006

GlobalParameters(
  Title = "This LSP was automatically generated by
CSP Converter at Mon Apr 24 17:04:54 2006"
  Version = "2.1"
  CRLHandlingMode = OPTIONAL
  LDAPLogMessageLevel = DEBUG
  SystemLogMessageLevel = DEBUG
  PolicyLogMessageLevel = DEBUG
  CertificatesLogMessageLevel = DEBUG
)
SyslogSettings(
  Server = 127.0.0.1
  Facility = LOG_LOCAL7
)
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 *= "C=RU,O=GINS,OU=QA,CN=MSCA"
  SerialNumber = "33ae6c6b85536f834a8d8e5358333f4f"
  Subject *= "C=RU,O=GINS,OU=QA,CN=MSCA"
)
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
DPDIIdleDuration     = 10
DPDResponseDuration = 3
DPDRetries           = 5
)
IPsecAction CMAP_1(
    TunnelingParameters *= TunnelEntry(
        PeerIPAddress = 10.0.110.101
        DFHandling=CLEAR
    )
    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 *= 192.168.101.0/24 )
    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 )
)
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 *= "iprb1"
    Action *= ( PASS )
)

```

Таблица маршрутов устройства GW1

Routing Table: IPv4

Destination	Gateway	Flags	Ref	Use	Interface
192.168.13.0	10.0.110.1	UG	1	2324	
192.168.101.0	192.168.16.254	UG	1	5	
192.168.16.0	192.168.16.1	U	1	62	iprb0
192.168.16.0	192.168.16.1	U	1	0	iprb0:1
192.168.16.0	192.168.16.1	U	1	0	iprb0:2
10.0.0.0	10.0.110.101	U	1	91	iprb0

10.0.0.0	10.0.110.101	U	1	0	iprb0:1
10.0.0.0	10.0.110.101	U	1	0	iprb0:2
default	10.0.110.254	UG	1	30	
127.0.0.1	127.0.0.1	UH	3	91931	lo0

Таблица маршрутов устройства GW2

Routing Table: IPv4

Destination	Gateway	Flags	Ref	Use	Interface
192.168.13.0	10.0.110.1	UG	1	83	
192.168.101.0	192.168.16.254	UG	1	1	
192.168.16.0	192.168.16.4	U	1	33	iprb1:1
10.0.0.0	10.0.110.104	U	1	63	iprb0:1
default	10.0.110.254	UG	1	14	
127.0.0.1	127.0.0.1	UH	2	1148	lo0

Таблица маршрутов устройства GW3

Routing Table: IPv4

Destination	Gateway	Flags	Ref	Use	Interface
192.168.103.0	192.168.103.127	U	1	16	iprb1
10.12.118.0	10.12.118.1	U	1	982	iprb0
224.0.0.0	10.12.118.1	U	1	0	iprb0
default	10.12.118.254	UG	1	2022	
127.0.0.1	127.0.0.1	UH	2	932	lo0

Вывод команды ifconfig –а устройства GW1

```
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 8
    inet 10.0.110.101 netmask ffff0000 broadcast 10.0.255.255
    ether 0:a0:c9:85:6f:17

iprb0:1: flags=1040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4> mtu
1500 index 8
    inet 10.0.110.102 netmask ffff0000 broadcast 10.0.255.255

iprb0:2: flags=1040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4> mtu
1500 index 8
    inet 10.0.110.103 netmask ffff0000 broadcast 10.0.255.255

iprb1: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 7
    inet 192.168.16.1 netmask ffffffff broadcast 192.168.16.255
    ether 0:e:a6:78:83:2d

iprb1:1: flags=1040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4> mtu
1500 index 7
    inet 192.168.16.2 netmask ffffffff broadcast 192.168.16.255
```

```
iprb1:2: flags=1040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4> mtu
1500 index 7
    inet 192.168.16.3 netmask ffffffff broadcast 192.168.16.255
```

Вывод команды `ifconfig` –а устройства GW2

```
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 8
    inet 10.0.110.101 netmask ffff0000 broadcast 10.0.255.255
    ether 0:50:fc:90:7f:1c
iprb0:1: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 8
    inet 10.0.110.104 netmask ffff0000 broadcast 10.0.255.255
iprb1: flags=1000842<BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 7
    inet 192.168.16.1 netmask ffffffff broadcast 192.168.16.255
    ether 4c:0:10:71:5c:b6
iprb1:1: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 7
    inet 192.168.16.4 netmask ffffffff broadcast 192.168.16.255
```

Вывод команды `ifconfig` –а устройства GW3

```
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 10.12.118.1 netmask ffffffff broadcast 10.255.255.255
    ether 0:40:f4:d8:43:fc
iprb1: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 3
    inet 192.168.103.127 netmask ffffffff broadcast 192.168.103.255
    ether 0:40:f4:d8:43:fb
```