grigoryev.victor@gmail.com vmg.pp.ua labs.mikrotik.com.ua

Григорьев Виктор

Днепропетровский национальный университет
Кафедра ЭВМ
Факультет физики, электроники и компьютерных систем

Виртуальная лаборатория для изучения компьютерных сетей

Введение

Виртуальная сетевая лаборатория - это соединённые между собой виртуальные машины, в которых запущены операционные системы сетевых устройств.

Для создания сетевых топологий, как правило, используется технология Drug-and-Drop: зацепил сетевое устройство мышью и перетащил его на рабочее поле. Устройства соединяются между собой каналами связи с помощью мыши.

Интерес к виртуальным лабораториям обусловлен:

- нехваткой реального сетевого оборудования для обучения студентов;
- невозможности давать домашние задания.

Трудности с определением производителя сетевого оборудования для реальной сетевой учебной лаборатории. Неприемлемы сетевые устройства, **обладающие**:

- ограниченной функциональностью;
- лишь Веб-интерфейсом для настройки.

Другие области применения сетевых лабораторий:

- подготовка к сертификации
- испытательный стенд при начальной стадии проектирования

Преимущества:

- отсутствие затрат на оборудование
- не надо места для развёртывания
- высокая скорость сбора и коммутации сетевой топологии
- каналы связи имеют 100% надёжность и заявленную пропускную способность

Недостатки: невозможно учесть реальные условия работы.

Поместив лабораторию на общедоступном сетевом ресурсе, с помощью удалённого рабочего стола можно обеспечить

- совместную работу над сетевым проектом
- коллективное обучение.

Не требуется организация одновременного присутствия людей в одном месте.

Дополнение к персональной виртуальной лаборатории.

Виртуальные сетевые лаборатории в обучении

Виртуальные лаборатории для изучения сетевых технологий в устройствах, произведенных фирмой Cisco, занимают доминирующее положение.

Для лабораторий Netsim фирмы Bosson и PacketTracer фирмы Cisco разработаны лабораторные практикумы для подготовки к сертификации CCNA и CCNP

Сетевая общественность для подготовки к сертификациям Cisco широко использует виртуальную машину Dynamips, поддерживающую широкий спектр операционных систем IOS. Эта машина запускается из командной строки и требует к себе бережного отношения. Для предотвращения 100%-й утилизации процессора необходимо подбирать параметр idlePC.

Этого недостатка лишена виртуальная машина командной строки IOU, работающая лишь с парой IOS. В комплект поставки IOU входит ряд топологий для подготовки к сертификации CCIE.

Dynamips входит в состав контейнера виртуальных машин **GNS3**. GNS3 – графическая оболочка, позволяющая работать с визуальным представлением сетевой топологии в режиме Drug-and-Drop.

В связке Dynamips-GNS3 сетевая топология хранится в текстовом файле, а конфигурации сетевых устройств присутствуют в файлах в виде команд IOS. Настроенная в одном GNS3 топология легко переносима в другой GNS3.

Есть ряд платных курсов по сертификациям CCNP и CCIE, которые базируются на Dynamips и GNS3:

1. CCIE Service Provider Workbook фирмы ieMentor;
2. CCIE Lab Workbook Internetworkexpert (ine.com) .

Эти курсы снабжены файлами конфигураций и топологий для Dynamips и GNS3.

Есть огромные репозитарии, хранилища топологий и конфигураций под Dynamips и GNS3.

http://www.gns3-labs.com
http://gns3vault.com
http://7200emu.hacki.at
gns3 workbench на http://sf.net

Для практически любой сетевой технологии можно найти реализацию в Dynamips и GNS3.

Всё это положительно влияет на продвижение продуктов Cisco.

Mikrotik RouterOs – лучший выбор для организации виртуальной сетевой лаборатории

Казалось бы связка Dynamips и GNS3 – идеальный выбор для организации учебной лаборатории. Опыт показал, что рядовой студент не относится к настройке параметра IdlePC в Dynamips с должной ответственностью и даже небольшая топология занимает всё процессорное время.

Помимо Dynamips в GNS3 входят виртуальные машины VirtualBox и Qemu. В Qemu можно запустить множество операционных систем и не только для Intel-платформы.

В качестве кандидатов в операционную систему для виртуальной лаборатории было опробованы JunOs от Juniper (FreeBSD), множество встроенных Linux (Vyatta, *WRT, Mikrotik RouterOs и др).

С учётом функциональности, быстродействия и потребляемым ресурсам выбор пал на

Mikrotik RouterOs

для Intel-платформы.

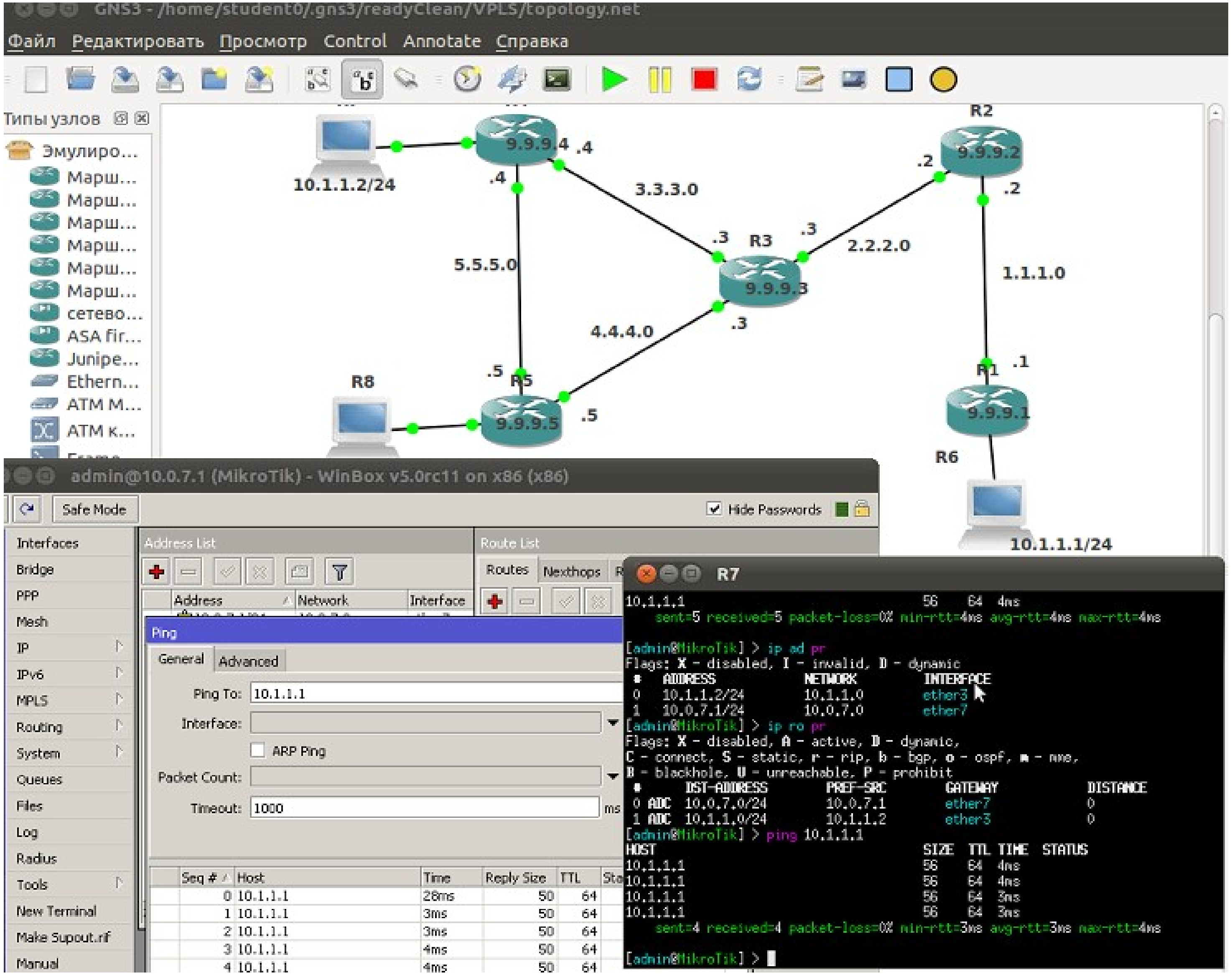

Рис.1. RouterOS в GNS3

Сколь угодно сложная топология из весьма большого числа RouterOs может быть запущена в qemu под GNS3 (Рис.1). Эту топология и конфигурацию устройств в ней можно сохранить и далее воспроизвести на другом GNS3.

GNS автоматически генерирует команды для Qemu, например для двух RouterOs. Ethernet моделируется UDP-каналом, консоль – с помощью протокола telnet, а связь RouterOs с хост-машиной осуществляется с помощью tap-интерфейсов

qemu -name R0 -m 256 /home/student7/projects/first/working/R0/FLASH -hdb /home/student7/projects/first/working/R0/SWAP -net nic,vlan=0 –net udp,vlan=0,sport=27002,dport=27003,daddr=127.0.0.1 -serial telnet:127.0.0.1:3700,server,nowait -net nic,vlan=6 -net nic -net tap,script=no,downscript=no,vlan=6,ifname=tap700

qemu -name R1 -m 256 /home/student7/projects/first/working/R1/FLASH -hdb /home/p/.gns3/first/working/R1/SWAP -net nic,vlan=1 -net udp,vlan=1,sport=27003,dport=27002,daddr=127.0.0.1 -serial telnet:127.0.0.1:3701,server,nowait -net nic,vlan=6 -net nic -net tap,script=no,downscript=no,vlan=6,ifname=tap701

.

Ubuntu – лучший выбор для хост-машины виртуальной сетевой лаборатории

В Linux **KVM** (Kernel-based Virtual Machine), поддерживающее аппаратную виртуализацию на базе процессоров Intel VT либо AMD SVM. Сам по себе KVM не выполняет эмуляции и используется совместно с виртуальными машинами. Мы будем использовать KVM без оптимизатора памяти ksmd.

Qemu под Windows не поддерживает KVM и при запуске нескольких экземпляров Qemu используется только одно ядро центрального процессора, что существенно замедляет работу с большими сетевыми топологиями. Разработчики Qemu винят в этом планировщик процессов Windows.

Рассмотрим влияние KVM на время загрузки в Qemu версии 13.0 в GNS3 нескольких экземпляров RouterOS версии 5.20 с памятью 64 Мб на компьютере с двуядерным процессором 2.4 ГГц и 4Гб памяти под управлением Ubuntu (табл. 1). Графики зависимости времени загрузки от числа экземпляров одновременно запущенных RouterOS приведен ра рис.2. RouterOS считается загруженной, если она выдала в консоли слово «MikroTik». Это проверяется с помощью следующего скрипта на язвке python

```
tn = telnetlib.Telnet()
for p in range(3000,3079):
    while 1:
        try:
            tn.open("127.0.0.1",p)
            tn.write("\r")
            tn.read_until("MikroTik")
            break
        except socket.error:
            print p
            time.sleep(1)
            continue
```

Время загрузки нескольких экземпляров RouterOS в GNS3 в Ubuntu.Табл. 1.

Кол-во экземпляров RouterOS N	Qemu без KVM		Qemu с KVM	
	Общее время загрузки T сек.	Время загрузки одного экземпляра N/T сек.	Общее время загрузки T сек.	Время загрузки одного экземпляра N/T сек.
16	82	5.125	30	1.875
32	166	5.1875	62	1.9375
48	260	5.416667	100	2.083333
64	470	7.34375	140	2.1875
80	Недостаточно памяти		180	2.25

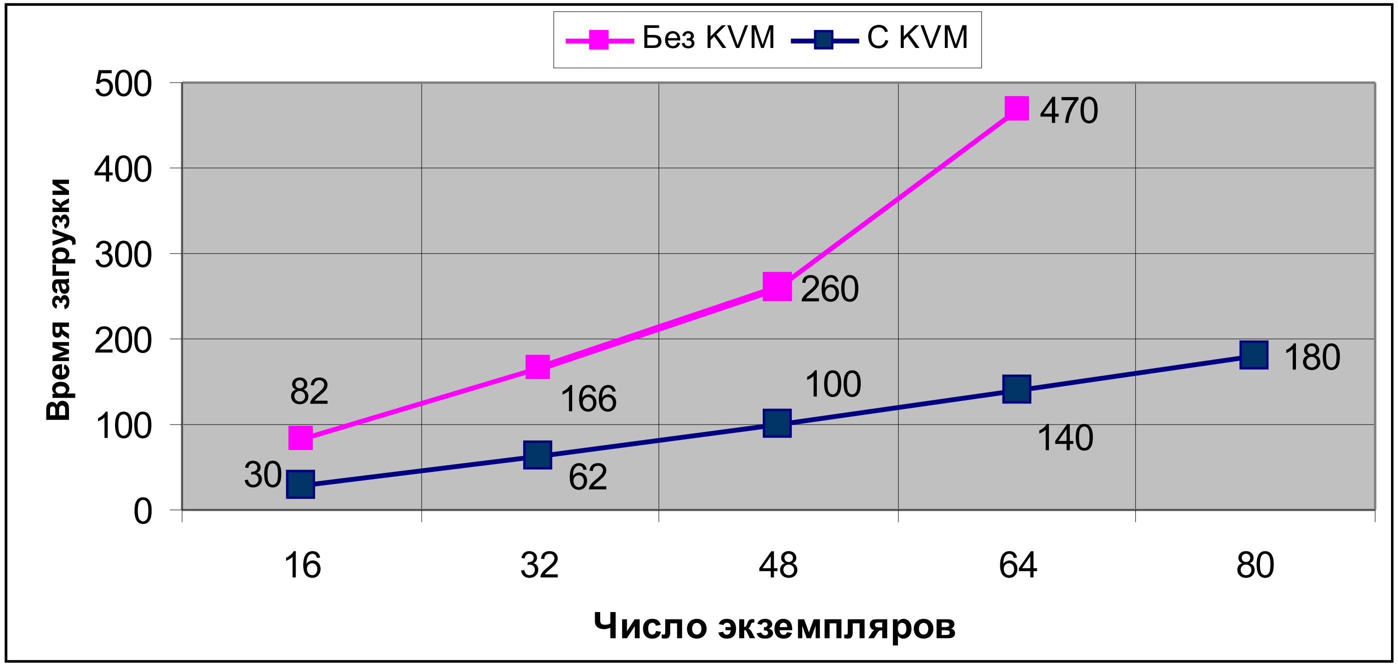

Рис. 2. Время загрузки нескольких экземпляров RouterOS в GNS3 в Ubuntu (2 ядра) .

Из табл. 1.1 видим, что KVM уменьшает время загрузки RouterOS более чем в два с половиной раза и это время составляет около двух секунд. Для обычного бытового компьютера имеем приемлемое время загрузки 80-ти экземпляров RouterOS равное 3 минутам.

Загрузим одновременно 32 экземпляра RouterOS (62с). Не останавливая загруженные RouterOS, запустим в новом GNS3 одновременно ещё 32 экземпляра RouterOS. На это уйдёт 75с. Итого 137с., что сравнимо с временем одновременного старта 64-х RouterOS (140с., согласно, Табл. 1.1).

Число одновременно работающих экземпляров RouterOs под Qemu определяется свободной памятью хост-машины Ubuntu. Анализ показал, что каждый экземпляр RouterOS с памятью 64 Мб, запущенный в Qemu с KVM, требует у Ubuntu 32 Мб памяти, а каждый экземпляр RouterOS с памятью 128 Мб запущенный в Qemu с KVM, требует у Ubuntu 54 Мб памяти.

Время загрузки нескольких экземпляров RouterOS в GNS3 в Ubuntu для компьютера с 4-х ядерным Intel i5-3550.

Табл. 1.1

Кол-во экземпляров RouterOS N	Qemu без KVM		Qemu с KVM	
	Общее время загрузки T сек.	Время загрузки одного экземпляра N/T сек.	Общее время загрузки T сек.	Время загрузки одного экземпляра N/T сек.
16	35	2.1875	23	1.4375
32	60	1.875	40	1.25
48	90	1.875	58	1.208333
64	120	1.875	73	1.140625
80	152	1.9	90	1.125

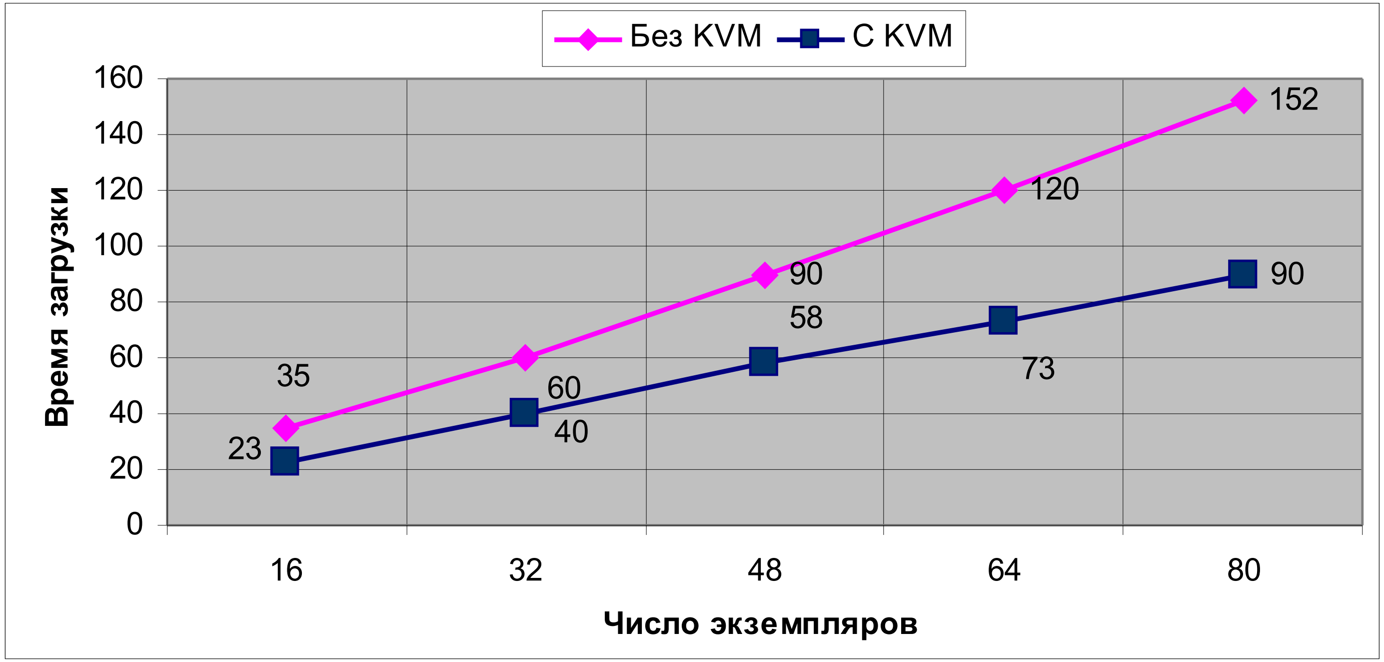

Рис. 2.1. Время загрузки нескольких экземпляров RouterOS в GNS3 в Ubuntu с 4-х ядерным Intel i5-3550.

Операционную систему Ubuntu можно запускать также под управлением виртуальной машины, например HYPER-V фирмы Microsoft. Qemu в Ubuntu под управлением HYPER-V работает медленнее, чем Qemu в Ubuntu на реальном компьютере. Это обусловлено, помимо прочего, и тем, что KVM не работает на виртуальной аппаратуре HYPER-V.

Хост-компьютер для HYPER-V имеет 4-ядерный процессор Intel Core i7 950 частотой 3066 МГц и 24Гб памяти. В HYPER-V запущена система Ubuntu с 12Гб памяти и 4-мя виртуальными ядрами.

Рассмотрим время загрузки в Qemu в GNS3 нескольких экземпляров RouterOS версии 5.20 с памятью 64 Мб на виртуальной Ubuntu (табл. 2). Графики зависимости времени загрузки в зависимости от числа экземпляров RouterOS приведен на рис.2.

Время загрузки нескольких экземпляров RouterOS в Ubuntu под HYPER-V . Табл. 2.

Кол-во экземпляров RouterOS N	Общее время загрузки T сек.	Время загрузки одного экземпляра N/T сек.
16	39	2.4375
32	75	2.34375
48	118	2.458333
64	155	2.421875
80	195	2.4375

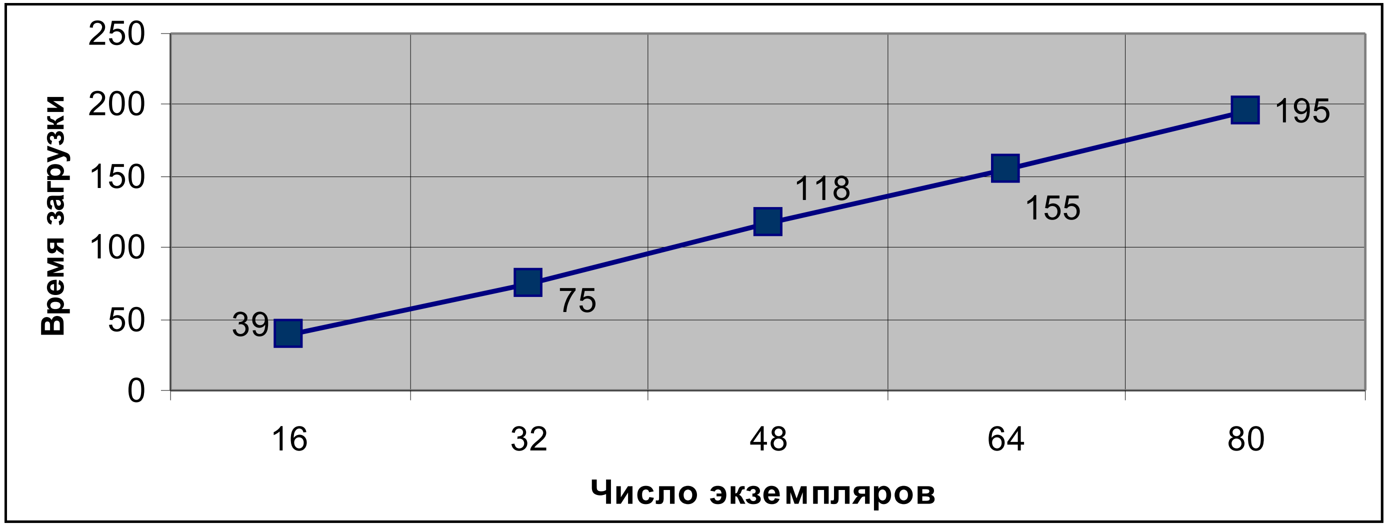

Рис. 2. Время загрузки нескольких экземпляров RouterOS в GNS3 в Ubuntu под HYPER-V (4 ядра).

Из табл. 1.2 видим, что время загрузки RouterOS под Qemu в Ubuntu под HYPER-V составляет около двух с половиной секунд, что сравнимо по времени с бытовым компьютером. Для виртуального компьютера имеем приемлемое время загрузки 80-ти экземпляров RouterOS равное 195 секундам.

Анализ показал, что каждый экземпляр RouterOS с памятью 64 Мб, запущенный в Qemu, требует у Ubuntu под HYPER-V 57 Мб памяти, а каждый экземпляр RouterOS с памятью 128 Мб запущенный в Qemu, требует у Ubuntu HYPER-V 79 Мб памяти. То есть RouterOS в Qemu с KVM потребляет приблизительно в двое меньше памяти, чем RouterOS в Qemu без KVM

Многоядерность процессоров распараллеливает загрузку нескольких экземпляров RouterOs. Так время (82с.) загрузки 16-ти экземпляров RouterOS на бытовом 2-х ядерном компьютером без поддержки KVM приблизительно в два раза больше, чем время (39с.) загрузки на виртуальном 4-х ядерном процессоре HYPER-V (см. табл. 1 и 2). То есть реальное ядра бытового компьютера имеют приблизительно одинаковую мощность по сравнению с ядром виртуального процессора HYPER-V.

Импорт и экспорт топологий

GNS3 хранит связи в сетевой топологии в текстовом виде, например, если интерфейс e0 роутера R0 связан с интерфейсом e1 роутера R1, то имеем

[[QEMU R0]]
e0 = R1 e1
[[QEMU R1]]
e1 = R0 e0

Для сохранения настроек предварительно обеспечивется IP-связь хост-машины с экземплярами RouterOs через tap-интерфейс. Здесь 10.0.$i.1 – адреса в RouterOs, связанные с tap-интерфейсом хост-машины.

Создание бинарной резервной копии. Скрипт makeGetBackup

```
#!/bin/bash
for ((i=$1;i<=$2;i++)) ;do
ssh admin@10.0.$i.1 "system backup save name=R$i"
sftp admin@10.0.$i.1:/R*
done
```

Пример резервного копирования 8-ми RouterOs с адресами 10.0.**0**.1 … 10.0.**7**.1

```
./makeGetBackup 0 7
```

Восстановление из бинарной резервной копии. Скрипт Restore

```
#!/bin/bash
for ((i=$1;i<=$2;i++)) ;do
scp R$i.backup admin@10.0.$i.1:
ssh admin@10.0.$i.1 "system backup load name=R$i"
done
```

Пример воостановления 8-ми RouterOs с адресами 10.0.0.1 … 10.0.7.1

./Restore 0 7

Скрипт makeGetExport для экспорта **команд RouterOS :**

```
#!/bin/bash
for ((i=$1;i<=$2;i++)) ;do
ssh admin@10.0.$i.1 "export compact file=E$i"
sftp admin@10.0.$i.1:/E*
done
```

Скрипт **Import** для импорта **команд RouterOS:**

```
#!/bin/bash
for ((i=$1;i<=$2;i++)) ;do
scp E$i.rsc admin@10.0.$i.1:
ssh admin@10.0.$i.1 "import E$i"
done
```

Использование аналогично: makeGetExport 0 7 и Import0 7

Не всё так гладко.

Например, экспортированная команда
routing bgp instance vrf add routing-mark=A redistribute-connected=yes

не импортируется, а требует добавки **instance=default**

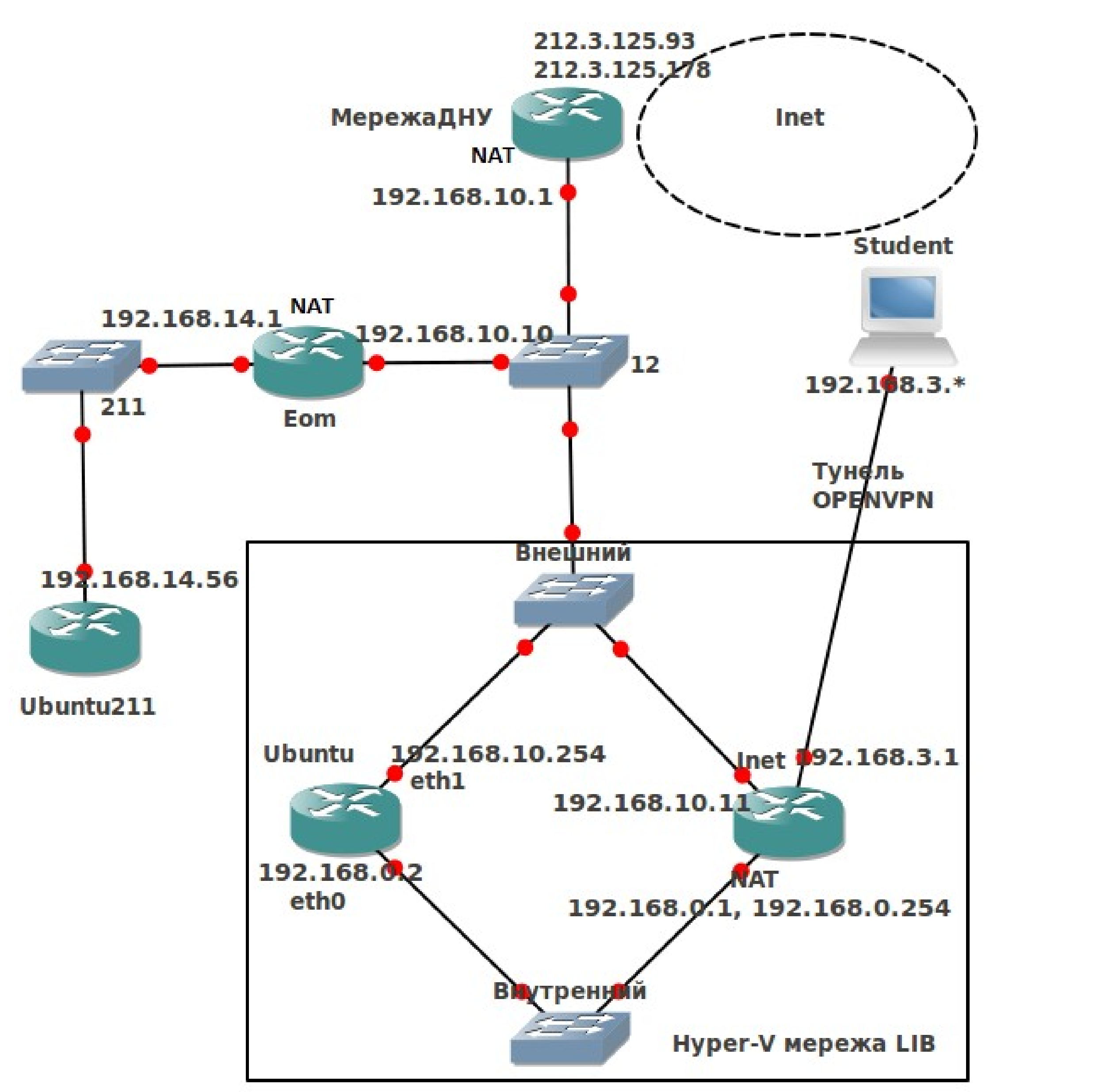

Рис.4. МережаДНУ – сеть ДНУ; Inet – Интернет; 211 – свич в ауд. 211; Eom – маршрутизатор в ауд. 211; 12 – свич 12-го уч. корпуса; Student – домашний компьютер студента; Ubuntu211 – Ubuntu в ауд. 211; Outer и Inner – внешний и внутренний свичи Hyper-V сети компьютера Lib; Ubuntu – виртуальная Ubuntu в Hyper-V компьютера Lib; eth0, eth1 – сетевые адаптери Ubuntu; Inet, NAT – – сетевые адаптери Hyper-V; Hyper-V мережа LIB – сетевые функции компьютера Lib.

Виртуальная сетевая лаборатория для коллективной работы

организована путём предоставления доступа к удалённому рабочему столу хост-компьютера под управлением Ubuntu, на котором развёрнута лаборатория. Используя импорт и экспорт топологий, студент может совместить работу на своём домашнем компьютере с удалённой работой.

В университете под лабораторию используется два Ubuntu-компьютера. Один реальный, а второй виртуальный d составе виртуальной машины Hyper-V под управлением Windows 2008 R2 . Место этих компьютеров в сети университета можно видеть на рис.4.

Компьютеры не имеют выхода в Интернет, поэтому для доступа к ним используется OpenVPN, организованный с помощью RSA-сертификатов.

OpenVPN позволяет организовать маршрутизацию между удалённым маршрутизатором Mikritik (в Qemu в GNS3) и домашним компьютером. Это позволит запустить на локальной машине Winbox для настройки удалённого маршрутизатора.

Для доступа из Windows к удалённому рабочему столу Ubuntu можно использовать обычный X-сервер для Windows, например Xming. Опыт показал, что этот подход приемлем только для работы в локальной сети. Может быть организована удалённая работа и по протоколу VNC, но при этом требуется множество дополнительных настроек.

Мы используем наиболее продвинутую технологию фирмы **NoMachine**, основанную на X-протоколе и протоколе SSH. Для удалённого доступа к рабочему столу Ubuntu следует выкачать с сайта производителя программу **NX Client for Windows** и установить её.

Каждый работающий удалённо получает свой набор tap-интерфейсов для связи Routeros с Ubuntu.

```
#!/bin/sh -e
taps="0 1 2 3 4 5"
ifs="00 01 02 03 04 05 06 07 10 11 12 13"
for s in $taps; do
for i in $ifs; do
openvpn --mktun --dev tap$s$i
brctl addbr m$s$i
brctl addif m$s$i tap$s$i
ifconfig tap$s$i up
ifconfig m$s$i 10.$s.$i.2 netmask 255.255.255.0 up
done
done
exit 0
```

Важно, чтобы у всех одновременно работающих не было общих окрытых портов TCP и UDP. Возьмём порты

10525+N - Qemuwraper

20000+N*1000 -UDP

3000+N*100 – консоль

Разработанные лабы

- Настройка типовой сети: DNS, DHCP, NAT, ARP, Firewall.
- Построение VPN второго и третьего уровня с помощью производных от PPP протоколов и OpenVPN
- IPsec-VPN типа сайт-сайт
- LDP (либо BGP) VPLS с организацией LSP с помощью LDP (либо RSVP)
- MPLS VPN 3-го уровня с организацией LSP с помощью LDP (либо RSVP)

Лаборатории развёрнуты:

vmg.pp.ua– на площадке Днепропетровского национального университета

labs.mikrotik.com.ua - Ультратех, ЛТД, Николаев – официальный дистрибьютор компании MikroTik в Украине (Дмитрий Лукин)

Подробное описание и конфигурации:
http://vmg.pp.ua/books/vnl
http://labs.mikrotik.com.ua

Выводы

Представляется целесообразным

- организовать сетевой репозитарий сетевых конфигураций RouterOS в виде топологий для GNS3
- использовать GNS3 в учебных курсах Mikrotik
- дистанционное обучение с управлением удалённым рабочим столом