К созданию виртуальных полигонов для исследования распределенных компьютерных систем

Андрей Зензинов
Ленар Сафин

Научные руководители:
д.ф-м.н. В. А. Васенин
к.ф-м.н. К. А. Шапченко

Мех-мат МГУ,НИИ механики МГУ

Таруса, 16 ноября 2012 г.

План доклада

- Введение
- Актуальность
- Постановка задачи
- Возможности автоматизации
- Требования к системе развёртывания
- Предлагаемый способ развёртывания
- Эксперименты
- Моделирование каналов связи

Введение

Направление: моделирование различных технологий на распределённых системах, в частности средств обеспечения информационной безопасности на grid и cloud-системах.

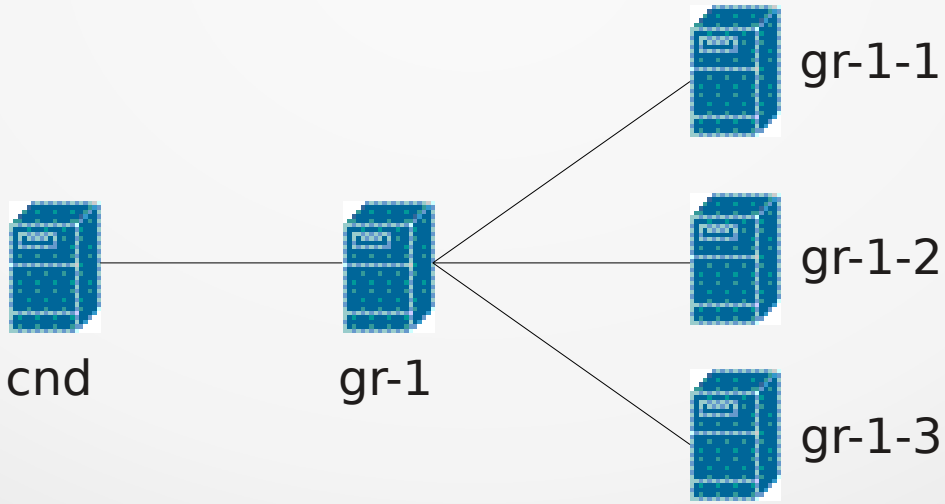
Пример: грид-система на основе Globus Toolkit.

Система подвергается атакам различного вида: DDoS, ARP-Spoofing, DNS-Spoofing, различные эксплойт-атаки и др.

Примеры

Пример использования. Grid-макет.

Пусть стоит задача развернуть грид-систему из узла-шлюза, набора локальных узлов и узла, распределяющего задания.



Разработка ПО и средств безопасности

В рамках решаемой задачи необходимо учитывать следующие факторы:

- Поведение системы в различных условиях
- Использование различных систем безопасности
- Варианты размещения злоумышленника относительно системы
- Различные архитектурные конфигурации

В итоге накапливается большая серия экспериментов.

Разработка ПО и средств безопасности

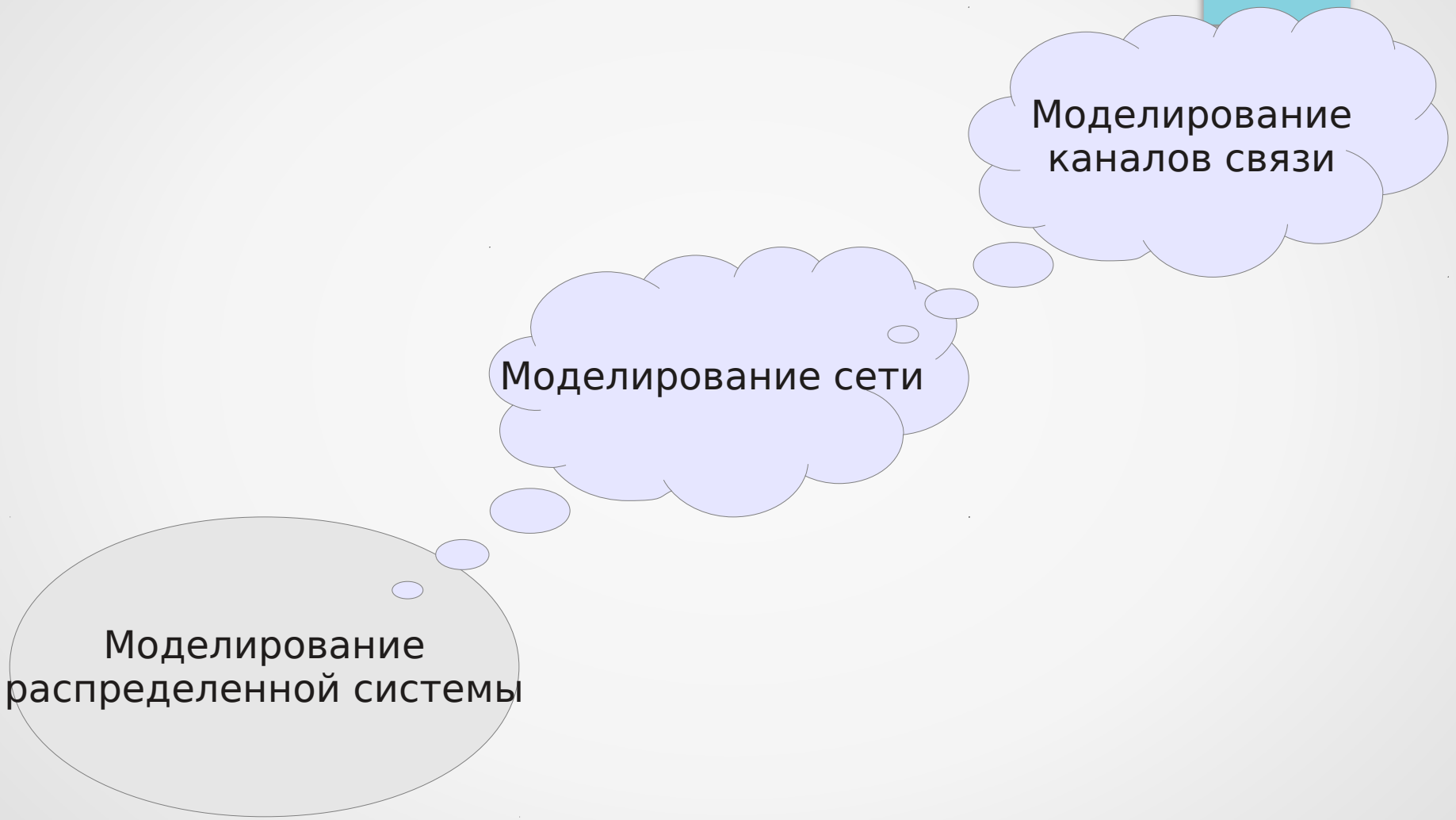
Возникает необходимость перестраивания системы.

Процесс создания системы:

- Создание набора узлов
- Установка ОС (на каждом узле)
- Настройка параметров системы (на каждом узле)
- Установка и настройка дополнительного ПО (на каждом узле)

Нужно автоматизировать этот процесс.

Моделирование каналов связи



Особенности реальных сетей

- Различные скорости передачи данных
- Пакетная фильтрация внутри сети
- Задержки, дублирование пакетов, потери
- Другие неполадки сети (например, разрыв соединения)

Постановка задачи

Разделим задачу создания виртуальных полигонов на две подзадачи:

- Автоматизация процесса развёртывания виртуального полигона
- Автоматизация внедрения средств моделирования каналов связи

Постановка задачи

Автоматизировать процесс развёртывания и настройки виртуального макета по определённой конфигурации.

Дополнительные свойства:

- Поддержка разных типов узлов
- Доступность этих узлов для выполнения заданий
- Используемое ПО должно быть открытым

Необходимость автоматизации

Должна быть возможность задавать различные сетевые и архитектурные конфигурации.

Процесс создания макета требует от оператора действий по заданному алгоритму для каждого узла.

Часть этих действий связана с ожиданием (например, копирование дискового образа, установка пакетов).

Оператор может ошибиться.

Необходимо автоматизировать этот процесс.

Процесс развёртывания

Рассмотрим 4 типа работ при развёртывании макета:

1. Создание набора ВМ
2. Установка ОС
3. Настройка параметров системы (в частности, сетевых)
4. Установка и настройка дополнительного ПО

Пути автоматизации

1. Использование инструментов типа libvirt и автоматизации скриптами
2. Препятствия:

   - Интерактивность установки дистрибутивов.

   Решения:

   - Использование различных решений вида debootstrap, kickstart-установка, файлы с ответами.

   - Использование клонирования ВМ.
3. Использование сетевых конфигураций и автоматическое редактирование.
4. Препятствия: автоматизация настройки для каждого отдельного программного комплекса должна задаваться отдельно (их много, все не укажешь).

Требования к макету

- Использование универсальных конфигураций
- Использование шаблонных ВМ
- Возможность создания набора копий шаблонной ВМ
- Автоматизация инициализации
- Возможность использовать как подготовленные дисковые образы, так и автоматическую установку
- Возможность ручного изменения и управления

Клонирование

Имеет смысл устанавливать ОС только на шаблонную ВМ, остальные же экземпляры создавать с помощью клонирования и некоторых изменений в настройках.

- Полное клонирование – создание полных копий дисковых образов
- Инкрементальное клонирование – клонирование только изменяющихся данных (настройки сети, ключи ssh)

Требования к конфигурации

- конфигурация должна перечислять все необходимые типы ВМ с указанием количества создаваемых копий
- указывать параметры ВМ для каждого типа (выделяемые ресурсы, адрес дискового образа, настройка и инициация)
- указывать общие параметры виртуализации
- описывать сетевые настройки
- описывать устанавливаемое после установки ПО

Пример конфигурации

```
{
"type" : "kvm",
"machines" : {
      "gw" : {
            "number" : 1,
            "memory" : "524288",
            "disk" : "/mnt/1.img"
         },
      "node" : {
            "number" : 8,
            "memory" : "130000",
            "disk" : "/mnt/2.img"
         }
   },
"network" : "Network 1.cfg"
}
```

Предлагаемый способ развёртывания

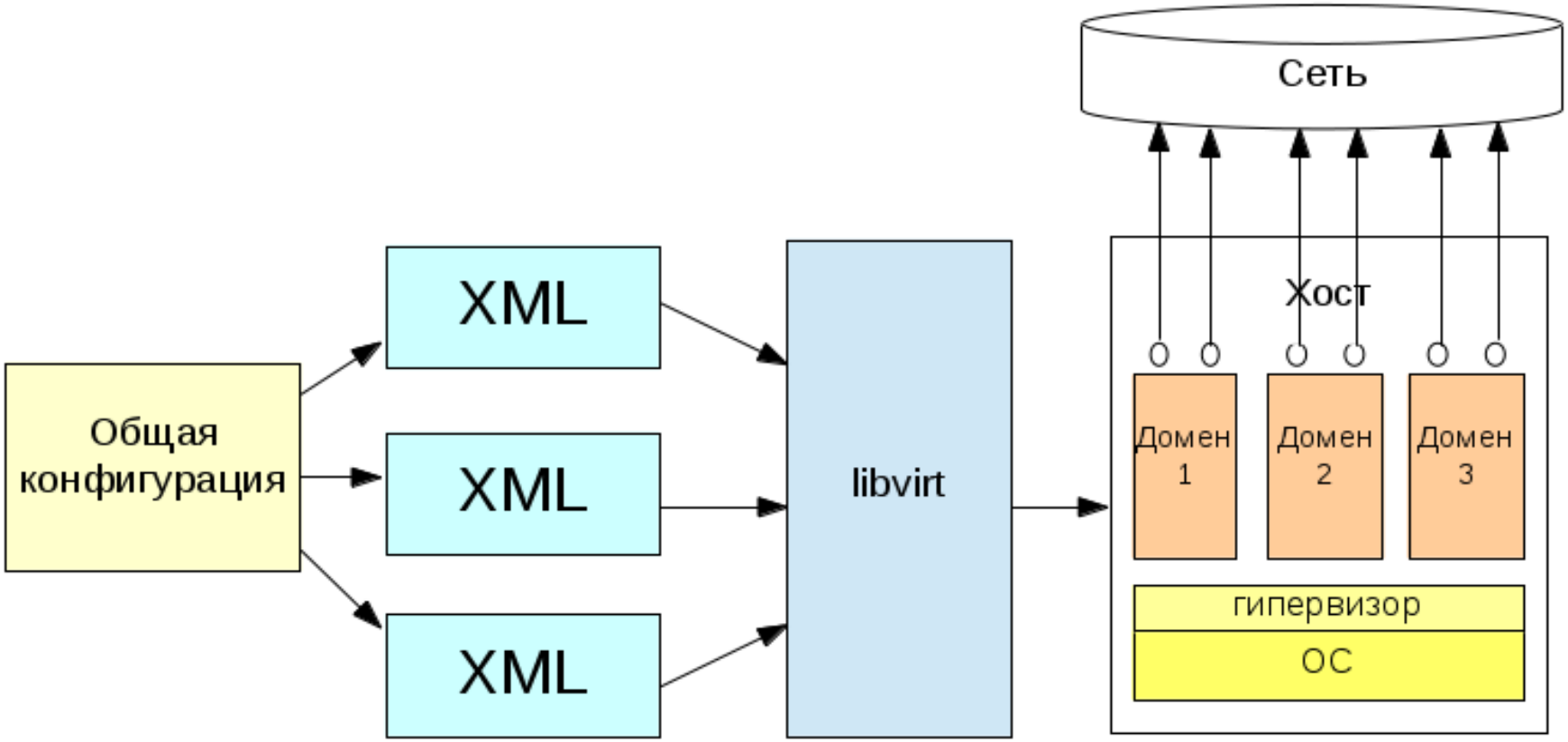
На данный момент нами создана автоматическая система развёртывания ВМ на основе библиотеки libvirt.

Поддерживается развёртывание макета из ВМ различных типов.

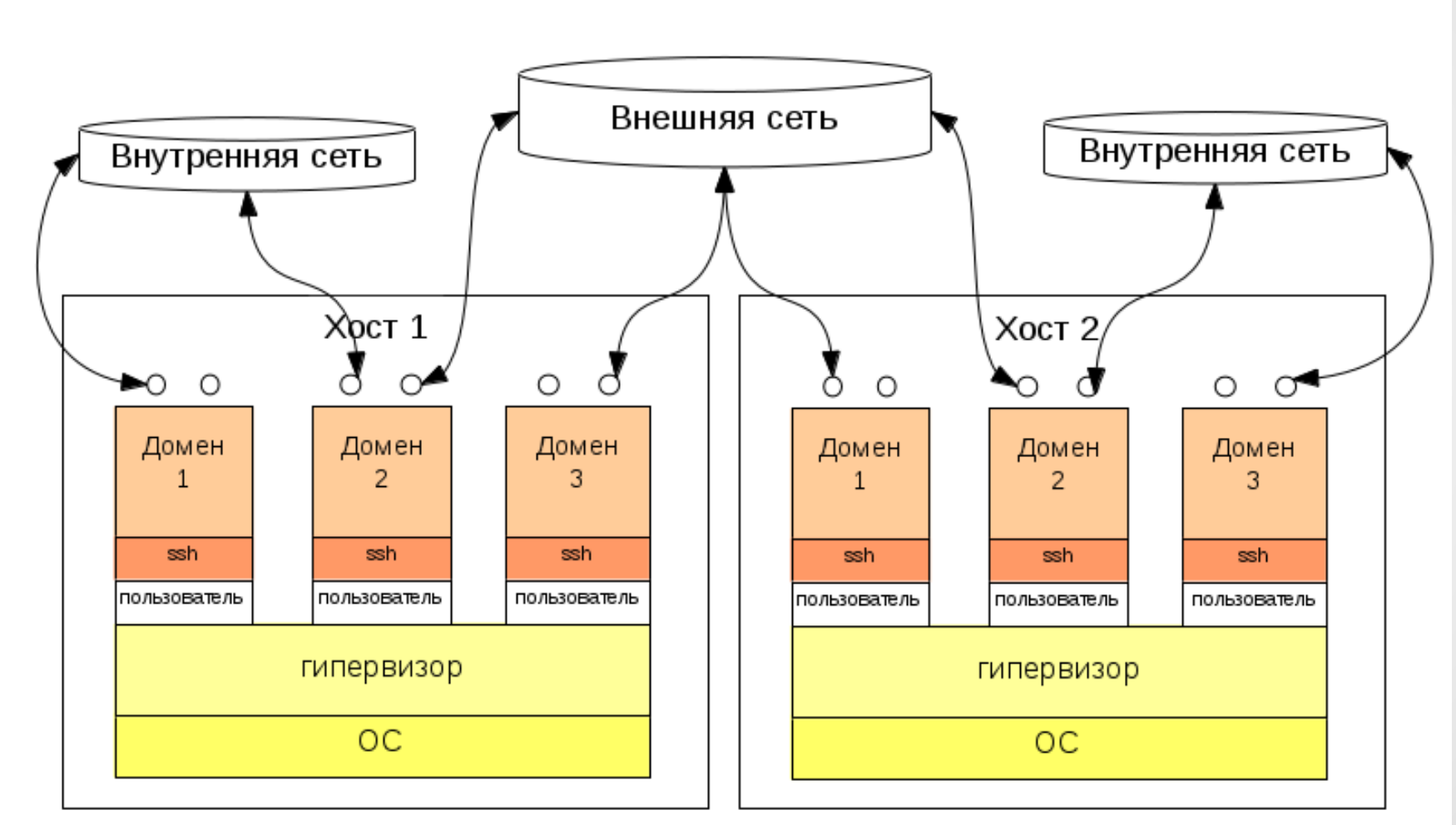
Общая схема работы такова:

- задаётся универсальная конфигурация в формате JSON
- задаются существующие дисковые образы шаблонных ВМ
- создаются инкрементальные копии шаблонных образов, настраиваются конфигурации сети и ssh
- создаётся XML-описание для каждого экземпляра ВМ
- через libvirt создаются и запускаются ВМ

Предлагаемый способ развёртывания



Предлагаемый способ развёртывания



Предлагаемый способ развёртывания

- Пользователь вручную задаёт только конфигурации и создаёт шаблонные ВМ, всё остальное выполняется автоматически.
- Имеется удалённый доступ к созданным ВМ через ssh.

Эксперименты

Пример:

DDoS-атака на узел 192.168.122.2

login.list:

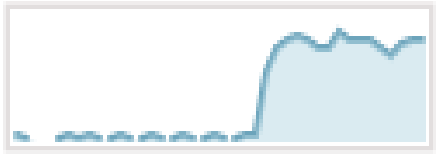
ssh -i vm1_rsa -p 2222 user@192.168.122.3
ssh -i vm2_rsa -p 2222 user@192.168.122.4
ssh -i vm3_rsa -p 2222 user@192.168.122.5
ssh -i vm4_rsa -p 2222 user@192.168.122.6

На управляющей системе запускаем

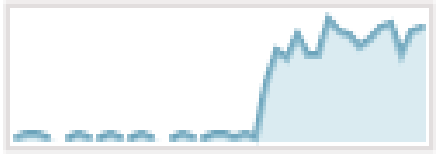
#parallel --tag --nonall --sshloginfile login.list sudo hping --flood 192.168.122.2

Эксперименты

node0
Работает

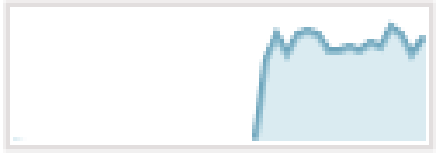


node1
Работает

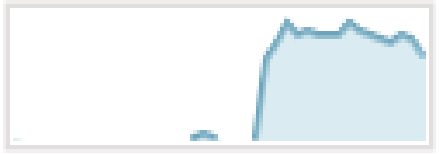


node2
Работает

node3
Работает



node4
Работает



Эксперименты

```
From 192.168.122.2 icmp_seq=2 Destination Host Unreachable
From 192.168.122.2 icmp_seq=3 Destination Host Unreachable
From 192.168.122.2 icmp_seq=4 Destination Host Unreachable
From 192.168.122.2 icmp_seq=5 Destination Host Unreachable
From 192.168.122.2 icmp_seq=6 Destination Host Unreachable
From 192.168.122.2 icmp_seq=7 Destination Host Unreachable
From 192.168.122.2 icmp_seq=8 Destination Host Unreachable
From 192.168.122.2 icmp_seq=9 Destination Host Unreachable
From 192.168.122.2 icmp_seq=10 Destination Host Unreachable
From 192.168.122.2 icmp_seq=11 Destination Host Unreachable
From 192.168.122.2 icmp_seq=12 Destination Host Unreachable
From 192.168.122.2 icmp_seq=13 Destination Host Unreachable
From 192.168.122.2 icmp_seq=14 Destination Host Unreachable
From 192.168.122.2 icmp_seq=15 Destination Host Unreachable
From 192.168.122.2 icmp_seq=16 Destination Host Unreachable
From 192.168.122.2 icmp_seq=17 Destination Host Unreachable
From 192.168.122.2 icmp_seq=18 Destination Host Unreachable
From 192.168.122.2 icmp_seq=19 Destination Host Unreachable
From 192.168.122.2 icmp_seq=20 Destination Host Unreachable
From 192.168.122.2 icmp_seq=21 Destination Host Unreachable
^C
--- 192.168.122.1 ping statistics ---
22 packets transmitted, 0 received, +21 errors, 100% packet loss, time 22338ms
pipe 4
[root@node0 ~]# _
```

Эксперименты

На системе с процессором Intel Core i5 и 16 GB RAM было запущено 200 ВМ

Наблюдения:

- Потребление памяти при создании – 14 GB
- Потребление памяти через некоторое время (1 час) – 7 GB
- Время развёртывания составило 24 минуты

Были использованы технологии UKSM и инкрементального клонирования

Автоматизация экспериментов

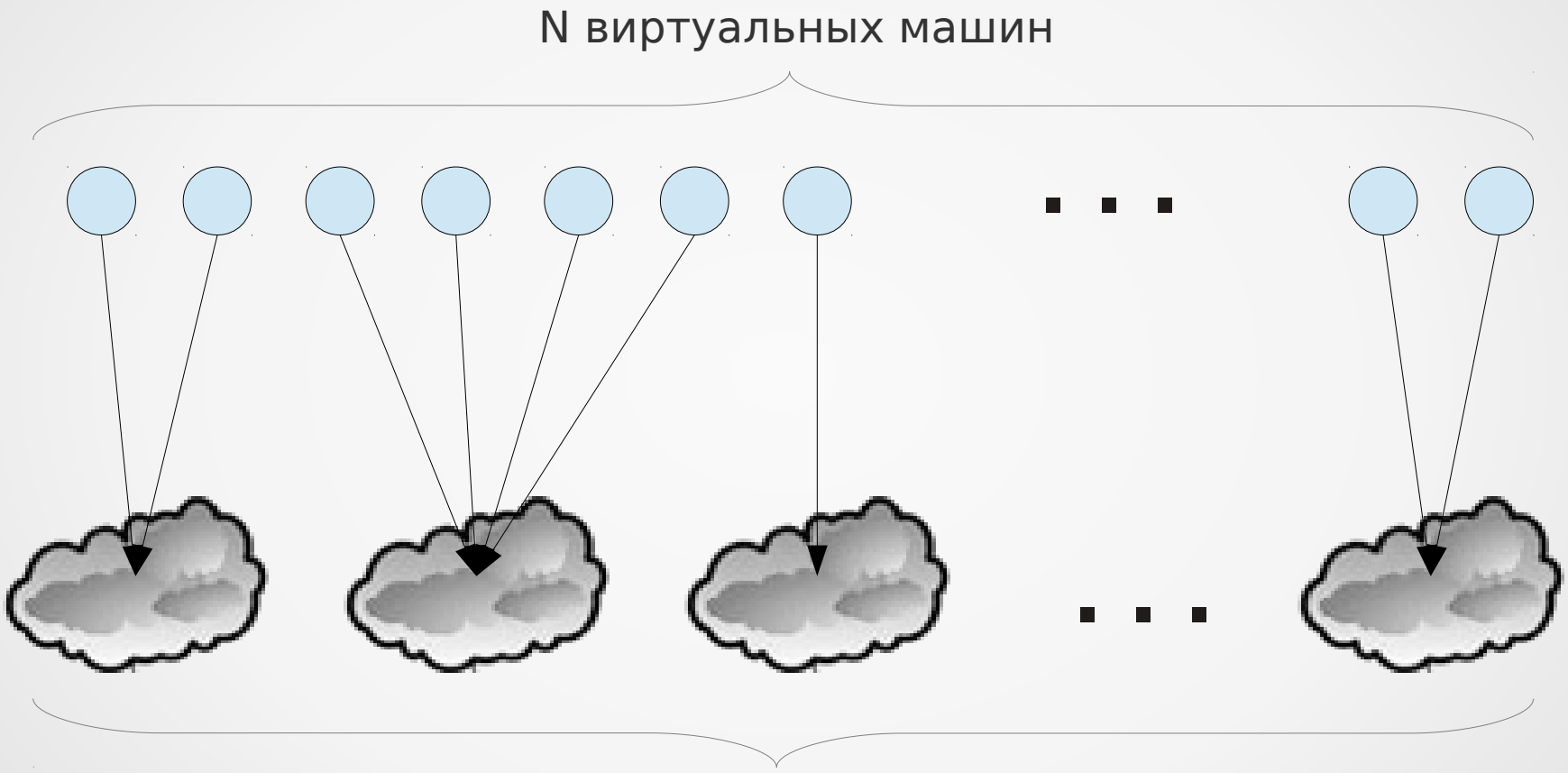
Можно задавать многие другие сценарии экспериментов

Использование параметров:

- Число узлов заданного вида
- Диапазон раздаваемых им адресов

Конфигурация системы может меняться, но используемый сценарий общий для всех

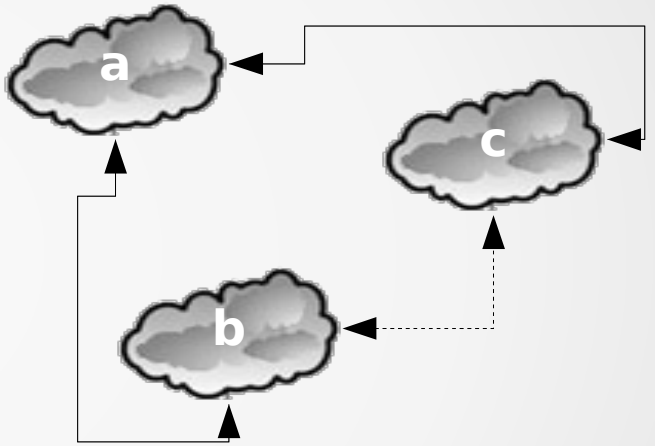
Постановка задачи (дано)



Постановка задачи

Необходимо:

- По заранее заданным связям, объединить подсети между собой
- Научиться моделировать каналы связи и маршрутизацию как между узлами внутри подсети, так и между подсетями
- Автоматизировать данный процесс
- Оптимизировать расходы машинных ресурсов на моделирование канала связи



Вариант решения задачи

Маршрутизация на промежуточном узле-роутере:

- Легко масштабируемо
- Слабые связи между узлами макета
- Возросшее потребление ресурсов
- Возможные проблемы, связанные с ограничениями на виртуализацию

Основные идеи моделирования канала связи

- Ширина канала (может быть разной на вход и на выход)
- Задержка распространения

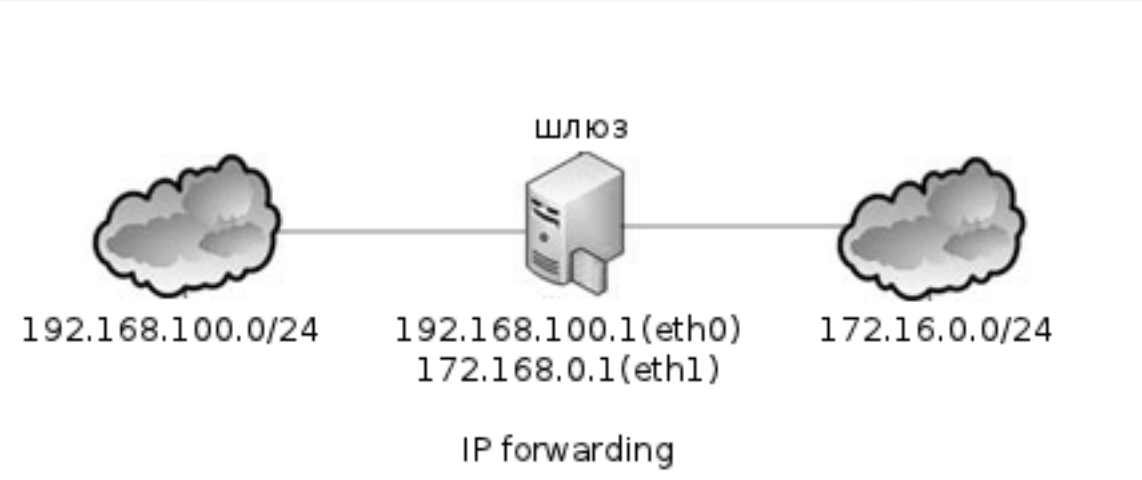
OPNET®
Making Networks and Applications Perform™

NetCracker®
Transforming the Service Layer™

comnet
Communication Networks

Объединение подсетей

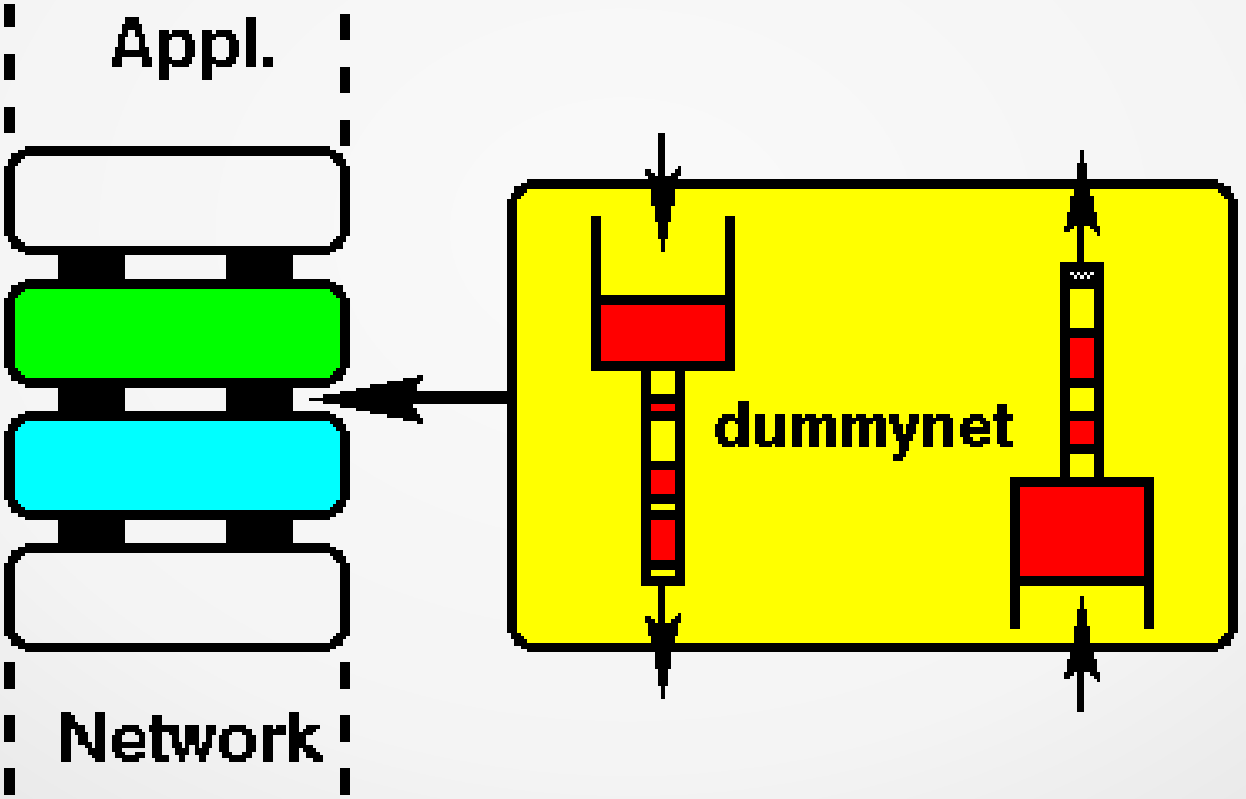
- Каждая сетевая карта виртуального маршрутизатора настраивается на подключение к одной из объединяемых подсетей
- В настройках маршрутизации узлов соответствующих подсетей маршрутизатор должен быть указан как шлюз
- IP Forwarding



Моделирование канала связи...

...на базе ОС FreeBSD

Ipfw + dummynet



Dummynet (примеры)

- Ограничиваем входящий TCP трафик до 2Mbit/s, а UDP до 300Kbit/s

  ipfw add pipe 2 in proto tcp

  ipfw add pipe 3 in proto udp

  ipfw pipe 2 config bw 2Mbit/s

  ipfw pipe 3 config bw 300Kbit/s

- Ограничиваем входящий трафик до 300Kbit/s для подсети 10.1.2.0/24

  ipfw add pipe 4 src-ip 10.1.2.0/24 in

  ipfw pipe 4 config bw 300Kbit/s queue 20 mask dst-ip 0x000000ff

- Имитация ADSL соединения с Луной:

  ipfw add pipe 3 out

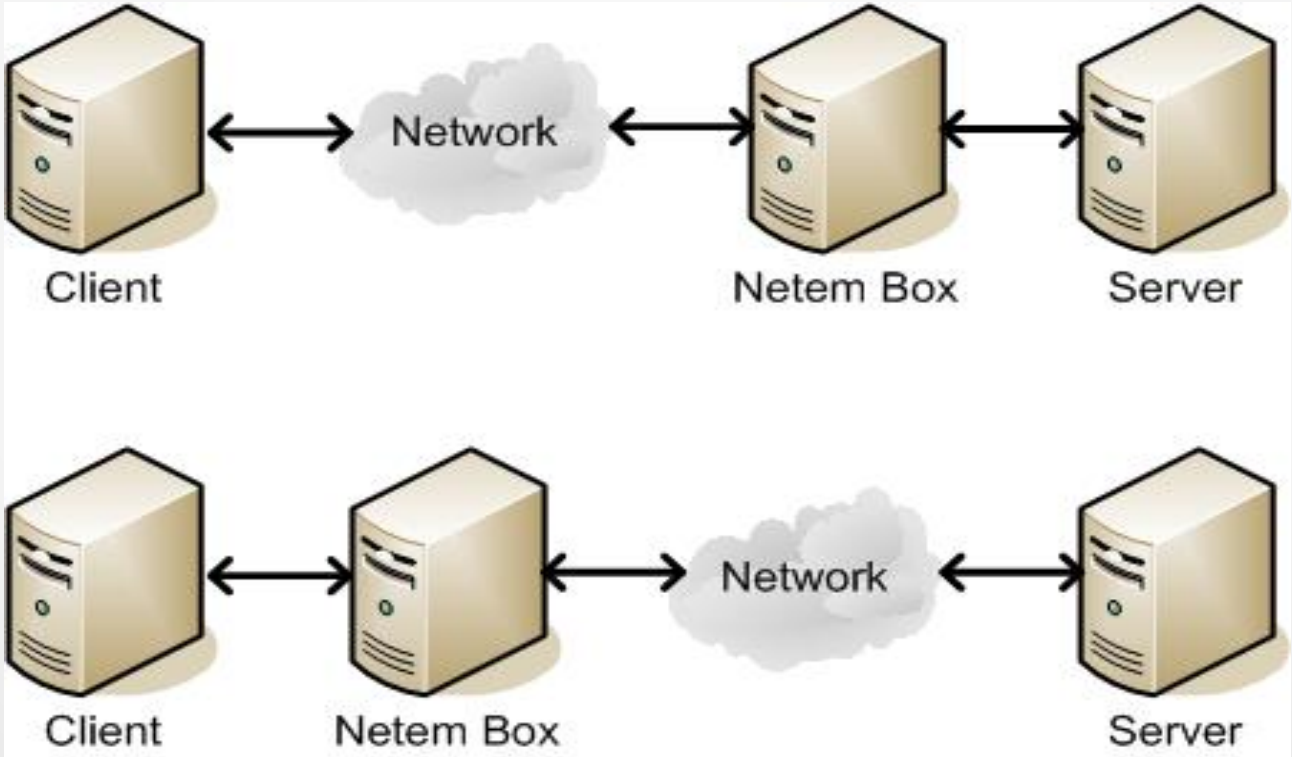
  ipfw add pipe 4 in

  ipfw pipe 3 config bw 128Kbit/s queue 10 delay 1000ms

  ipfw pipe 4 config bw 640Kbit/s queue 30 delay 1000ms

...на базе ОС Linux

HTB.Init/netem + iptables



Netem (пример)

Netem — модуль ядра Linux. Пример ниже — эмуляция 3G соединения:

```
modprobe netem
tc qdisc add dev $iface root handle 1:0 netem delay 100ms
tc qdisc add dev $iface parent 1:1 handle 10: tbf rate 512kbit
tc qdisc change dev $iface root netem loss 1%
tc qdisc change dev $iface root netem corrupt 0.5%
tc qdisc change dev $iface root netem duplicate 0.1%
```

Дальнейшие перспективы

- Расширение поддержки реализуемых систем:
  - Поддержка задаваемых сетевых конфигураций
  - Поддержка развёртывания ПО для распределённых систем (Globus Toolkit, MPI)
- Поддержка распределённой инфраструктуры для развёртывания ВМ
- Поддержка средств автоматизации настройки

Дальнейшие перспективы

- Автоматизация создания и развертывания виртуального маршрутизатора
- Создание своего мини-дистрибутива для виртуального маршрутизатора (для локального использования)

Спасибо за внимание!