САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Факультет Компьютерных технологий и управления

Кафедра вычислительной техники

ОТЧЕТ

о практике

Исследование сетевых технологий с помощью имитационного моделирования

Студент

Елькин А. А. группа 3105

Руководитель практики

Соснин В. В. доцент

2014 год

Содержание

1 Система компьютерной верстки TEX(LATEX)

1.1 TEX

1.1.1 История TEX

TEX [5] — система компьютерной верстки, разработанная Дональдом Кнутом, которая предназанчена для компьютерной верстки текста и математических формул. Кнут начал разрабатывать систему в 1977 году, и первая версия TEX вышла 1979 года. В 1982 году вышла заново переписанная версия TEX'a, которой было дано название TeX82. И с версии TEX 3.0, которая получила лучшую поддержку 8-битных символов и различных языков, используеться нумирация: каждое обновление добавляет в конец номера версии десятичную цифру так, что бы она приблежалась к числу π.

1.1.2 Особенности TEX

В TEX пользователь пишет тескст и задает лишь струкуту самого текста, а система сама формирует документ на основе выбранного шаблона. Для задания структуры используеться собственный язык разметки TEX'a, все это содержиться в фалье с расширением .tor, и TEX транслирует в файл .dvi.

TEX можно использовать для создания разных видов докуметов: книги, статьи, отчеты, письма и др.

1.2 LATEX

LATEX [6] — макропакет компьютерной верстки TEX. Он не добаляет возможности в TEX, а лишь позволяет автоматизировать задачи набора текста(умерация разделов и формул, перекресные ссылки, размещение таблиц и т. п.). Первую версию выпустил Лесли Лэмпортв 1984 году. В 1994 году была выпущена вторая версия LATEX– LATEX 2_ε, которая являеться текущей по сей день.

1.3 Достоинства и недостатки [1]

Среди достоинств можно выделить:

- Автор может не вникать в детали оформления документа, ему лишь надо задать логическую структуру текста.
- высокое качество и гибкость верстки абзацев и математических формул.

- TEX не требует большой вычеслительной мощности.
- Система работает на большенства платформах.

Среди недостатков можно выделить:

- Исходный текст не будет выглядеть так же как при печати.
- Создание нового макета документа очень трудоемкая задача.
- TEX плохо приспособлен для верстки страниц со сложным взаимодействие текста и графиков.

1.4 Список выбранного ПО

Для написания отчета по практике был выбра ряд программного обеспечения:

1. Сборка TEX'а MacTeX(http://tug.org/mactex/), включающий pdfLaTeX, который выдает документ с расширение pdf.
2. IDE Eclipse(https://www.eclipse.org/) с расширением TeXlipse(http://texlipse.sourceforge.net/), позволяющие удобно редактировать документ.

2 Система контроля верси Git

2.1 История Git

Git [7] — распределенная система контроля версиями. Причиной создания Git послужило ухудшение отношейний между сообществом разработчиков Linux и компанией разработавшей BitKeeper, используемым сообществом с 2002 года для разработки Linux. Создателем проекта был Линус Торвальдс, и на сегодняшний день поддердживаеться Джунио Хамано.

2.2 Особенности Git

В отличии от дугих СКВ Git не хранит изминения файлов, а сохраняет слепок файла как она выгляид в данный момент, при чем, если файл не был изменен, он делает ссылку на ранюю сохраненную версию файла. Так же большенство опраций с файлами происходит локально, то есть для просмотра истории изменения проекта, создания коммита можно не иметь доступа к Сети.

Git следит за целостностью данных, он вычисляет контрольную сумму(SHA-1 хеш), которая становиться индексом данного файла. Данная система не позваляет изменять содержимое файлов или каталога.

2.3 Git-команды[2]

1. git help предаставляет список команд. Если использовать git –help <имя команды> даст справку об определенной команде.
2. git init создает каталог .git со всей необходимой информацией о репозитории.
3. git clone клонирует существующий репозиторий
4. git remote отображает уже подключенные репозитории
5. git remote add добавляет удаленный репозиторий
6. git remote rm удаляет ссылку на репозиторий
7. git add индексирует измененных файлом
8. git commit фиксирует измененные файлы
9. git rm удаляет файлы

Примеры ипользования приведенных команды находяться в каталоге "примеры Git команд"

3 WiMAX

WiMAX[8] — телекоммуникационная технология, предоставляющая универсальный беспроводной связи на большие расстояния для разных устройств. Сеть WiMAX представляет собой совокупность беспроводного сегмента, который описывается в стандарте IEEE 802.16, и базового сегмента, определенный спецификациями WiMAX-форума[9].

3.1 История WiMAX

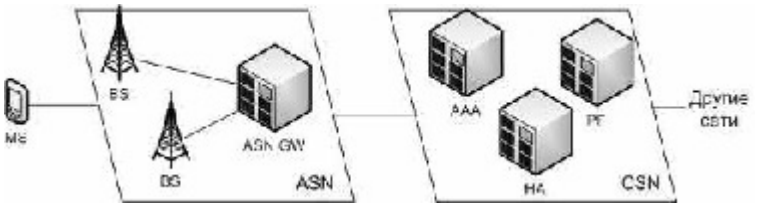
В декабре 2001 года была принята первая версия стандарта IEEE 802.16, который описывал организацию широкоплосной беспроводной связи. Стандарт предусматривал скорость передачи информации 32-134 Мбит/с на расстояние в 2-5 км в радиоканалах шириной 20, 25 и 28 МГц.

Из-за того, что была необходимо построение беспроводной сети только в зоне прямой видимости, стандарт не получил широкого распространения. По этому в январе 2003 года был принято расширение 802.15a, который изменял используемые частоты на ча стоты в диапозоне от 2 до 11 ГГц. Это расширение должно было обеспечивать скорость передачи информации 1-75 Мбит/с на теоретическое возможное растояние 50км(в основном 6-9 км).

Стандарт IEEE 802.16e приняли в 2005 году. Он поддерживает мобильных абонентов и систему роуминга между сетями различных беспроводных стандартов. 802.16e позволял без разрыва сеанса переключаться между стандартами 802.11 и 802.16.

3.2 Структура[3] WiMAX

Сеть состоит из двух основных подсистем: Access Service Network(Сеть доступа) и Connectivity Service Network(Сеть обеспечения услуг).



Структура сети WIMAX

CSN определяеться как набор функций для абонента сети:

- Распределение адрессов между абонентами
- Доступ к сети Интернет

- Контроль доступа абонентов в сети
- Биллинг и межаператорское взаимодействие
- Обеспечение сервисов WiMAX
- Обеспечение аутенфикации, авторизации и аудита соединения

CNS пренадлежит оператору WiMAX, которому пренадлежат маршрутезаторы, серверы для авторизации, вутенфикации и аудита, базы данных пользователей, устройства преобразования сигнала.

ASN — набор сетевых элементов, организующие доступ в WiMAX сеть, выполняющие функции:

- Доступ абонентов сеть по радиосвязи
- Передача сообщений между CSN и абонентом для обеспечения аутенфикации, авторизации и аудита соединения
- Правление радиоресурсами
- Поиск абонента в сети
- Мобильность абонента

ASN представляет собой множество базовых станций, которые предоставляю доступ по радиосвязи по стандарту IEEE 802.16, и шлюза, предназначенного для объединения трафика от базовых станций и передачи его в CNS. ASN обязательно включает одну базовую станцию и один ASN-шлюз. Одной ASN могут пользоваться несколько провайдеров WiMAX(каждый со своей CNS).

Но так же не стоит забывать о большом количестве абонентов(мобильных станций). В качестве их могут быть мобтльные телефены, смартфоны, ноутбуки и другие устройства со встроенным или внешним адаптером.

4 Выполнение задания по теме

4.1 ПО ns-3

ns-3[10] — это дискретно-событийный симулятор сети, свободно распространяющийся под лицензией GNU GPLv2 license. Основное использование ns-3 фокусируебться на беспровобном или IP моделирование, которое включаетодели для Wi-Fi, WiMAX и LTE.

4.1.1 Установка ns-3

Для установки потребовалось: Mercurial(hg)(http://aragost.com/mercurial/)-система контроля версии. GNU Compiler Collection(http://gcc.gnu.org/).

Для начала нужно было было клонировать репозиторий, в котором размещались скрипты для скачивания и установки ns-3(скриншот downloadScripts.jpg).

hg clone http://code.nsnam.org/ns-3-allinone

Затем запускаелся скрипт для загрузки исходных файлов ns-3(скришот downloadFileToInstall.jpg)

./download.py -n ns-3-dev

После этого запускался скрип, запускающий сборку ns-3.(скриншот DoBuild.jpg)

./build.py

4.1.2 Настройка

Настройка компилятора осуществляеться при помощи команды(скриншот ns3ConfigGCC.jpg):

CXX=g++ ./waf configure

Настройка каталога для собранных проектов и включения учебных примеров:

./waf -d debug -o build/debug –enable-examples –enable-tests configure

4.2 Разбор первого примера[4]

Для разработки модели в ns-3 используеться язык программирования C++.

```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/applications-module.h"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("FirstScriptExample");

int main (int argc, char *argv[])
{
  Time::SetResolution (Time::NS);
  LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
  LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);

  NodeContainer nodes;
  nodes.Create (2);

  PointToPointHelper pointToPoint;
  pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
  pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));

  NetDeviceContainer devices;
  devices = pointToPoint.Install (nodes);

  InternetStackHelper stack;
  stack.Install (nodes);

  Ipv4AddressHelper address;
  address.SetBase ("10.1.1.0", "255.255.255.0");

  Ipv4InterfaceContainer interfaces = address.Assign (devices);

  UdpEchoServerHelper echoServer (9);
```

```
  ApplicationContainer serverApps = echoServer.Install (nodes.Get (1));
  serverApps.Start (Seconds (1.0));
  serverApps.Stop (Seconds (10.0));

  UdpEchoClientHelper echoClient (interfaces.GetAddress (1), 9);
  echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
  echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
  echoClient.SetAttribute ("PacketSize", UintegerValue (1024));

  ApplicationContainer clientApps = echoClient.Install (nodes.Get (0));
  clientApps.Start (Seconds (2.0));
  clientApps.Stop (Seconds (10.0));

  Simulator::Run ();
  Simulator::Destroy ();
  return 0;
}
```

При помощи *include* подключаются заголовочные файлы, в которых описаны нужные модули. На следующей строке *using namespace ns3;* обозначает то, что мы используем пространство имен ns3(ns3::).

Макрос *NS_ LOG_ COMPONENT_ DEFINE ("FirstScriptExample");* используется для последущей документации при помощи системы документации Doxygen.

*int main (int argc, char *argv[])* являеться главной функции и первой входной точкой программы, как для любой C/C++ программы. Это связано с тем, что для моделирования используеться язык программирования C++(Код ns-3 просто программа на C++).

NodeContainer nodes; объявляет объект типа NodeContainer, а *nodes.Create (2);* создает 2 узла и добавляет сылки на них в этот NodeContainer.

PointToPointHelper pointToPoint; pointToPoint.SetDeviceAttribute ("DataRate StringValue ("5Mbps")); pointToPoint.SetChannelAttribute ("Delay StringValue ("2ms")); Упрощает соединение и настройку PointToPointNetDevice и PointToPointChannel объектов. Так же устанавливает для устройства(объекта PointToPointNetDevice) атрибут "DataRate"значение 5Mbps и задержку в 2мс для каждого созданного канала(PointToPointChannel) впоследствии.

NetDeviceContainer devices; devices = pointToPoint.Install (nodes); позволяет настроить созданные нами узлы в начале программы и канал между

ними(Параметры: скорость передачи данных 5 Мбит/с с задержкой в 2 мс).

InternetStackHelper stack; stack.Install (nodes); дает TCP/UPD/IP функционал существующим узлам.

Ipv4AddressHelper address; address.SetBase ("10.1.1.0 "255.255.255.0"); создает объект Ipv4AddressHelper, которому сообщаем что он должен начать выделять адрессз сети 10.1.1.0 используя маску 255.255.255.0. *Ipv4InterfaceContainer interfaces = address.Assign (devices);* присваивает адресса устройствам.

UdpEchoServerHelper echoServer (9); создает серверное приложение которое ждет UDP-пакет и потом отправляет его обратно отправителю(Используеться прот 9). *ApplicationContainer serverApps = echoServer.Install (nodes.Get (1)); serverApps.Start (Seconds (1.0)); serverApps.Stop (Seconds (10.0));* устанавливает приложение на узел 2 и задает время начала и конца работы.

UdpEchoClientHelper echoClient (interfaces.GetAddress (1), 9); создает приложение которое посылает UDP-пакет и ждет пока придет его эхо. *echoClient.SetAttribute ("MaxPackets UintegerValue (1)); echoClient.SetAttribute ("Interval TimeValue (Seconds (1.0))); echoClient.SetAttribute ("PacketSize UintegerValue (1024));* настройка параметров этого приложение: отправляет 1 пакет, интервал между пакетами 1 с и размер пакета 1024 бита.

ApplicationContainer clientApps = echoClient.Install (nodes.Get (0)); clientApps.Start (Seconds (2.0)); clientApps.Stop (Seconds (10.0)); устанавливат приложение на узел 1, с началом работы через 2 секунды после начала симуляции, и конец после 10 секунд от начала симуляции.

Simulator::Run (); запускает симуляцию. *Simulator::Destroy ();* освобождает память от симултируемой модели.

4.3 Описание разработанных моделей

4.3.1 Первая версия модели

В файле wimaxv1.cc представлен код первой версии модели. В данной моделе представлены два узла: мобильная станция(абонент) и базовая станция(*ssNodes* и *bsNodes* соответсвенно) Между узлами сигнал передаеться по стандарту 802.16.

При помощи ns3::WimaxHelper создаем сетевые устройства. Затем даем tcp/udp/ip-функционал при помощи *InternetStackHelper*. Каждому устройству назначаем адресс(адресс сети 192.168.1.0/24) при помощи Ipv4AddressHelper. *IpcsClassifierRecord* и *ServiceFlow* позволяют организовать передачу по стандарту 802.16. И для проверки модели на узле базовой станции разместим серверное приложение которое будет отслеживать входящие пакеты, а на

узле мобильное станции разместим приложение клиенское приложение, которое будет отправлять 3 пакета размером 1024 бита и интервалом в 0.5 секунд.

Результаты выполнения программы:

```
TraceDelay TX 1024 bytes to 192.168.1.2 Uid: 7770 Time: 4
TraceDelay: RX 1012 bytes from 192.168.1.1 Sequence Number: 0 Uid: 7770
TXtime: +4000000000.0ns RXtime: +4019827983.0ns Delay: +19827983.0ns
TraceDelay TX 1024 bytes to 192.168.1.2 Uid: 8773 Time: 4.5
TraceDelay: RX 1012 bytes from 192.168.1.1 Sequence Number: 1 Uid: 8773
TXtime: +4500000000.0ns RXtime: +4515255027.0ns Delay: +15255027.0ns
TraceDelay TX 1024 bytes to 192.168.1.2 Uid: 9741 Time: 5
TraceDelay: RX 1012 bytes from 192.168.1.1 Sequence Number: 2 Uid: 9741
TXtime: +5000000000.0ns RXtime: +5021237259.0ns Delay: +21237259.0ns
```

192.168.1.2 - адресс базовой станции, 192.168.1.1 - адресс клиенской станции.

4.3.2 Вторая версия модели

Вторая версия модели представлена в файле wimaxv2.cc. Структура этой модели такая же, как и у первой версии, только для мобильной станции(абонента), была добавлена возможность передвижения, и были добавленны серверное и клиенское приложение.

При помощи *RandomRectanglePositionAllocator* и *RandomWaypointMobilityModel* задаем область где должна двигаться мобильная база(абонент) и скорость. *bsNodes.Get(0)->GetObject<MobilityModel>()->SetPosition(Vector(0,0,0));* и *ssNodes.Get(0)->GetObject<MobilityModel>()->SetPosition(Vector(100,0,0));* задают начальные точки расположения базовой и мобильной базы соответсвенно. Так же для првоерки двунаправленной передачи данных были добавлены *UdpEchoServerHelper* и *UdpEchoClientHelper*, работа которых была описана в в разборе первого примера.

Результаты выполнения программы:

```
POS: x=100, y=0
At time 2s client sent 1024 bytes to 192.168.1.2 port 99
At time 2.0179s server received 1024 bytes from 192.168.1.1 port 49153
At time 2.0179s server sent 1024 bytes to 192.168.1.1 port 49153
```

```
At time 2.02312s client received 1024 bytes from 192.168.1.2 port 99
TraceDelay TX 1024 bytes to 192.168.1.2 Uid: 5867 Time: 3
TraceDelay: RX 1012 bytes from 192.168.1.1 Sequence Number: 0 Uid: 5867
TXtime: +3000000000.0ns RXtime: +3018863773.0ns Delay: +18863773.0ns
TraceDelay TX 1024 bytes to 192.168.1.2 Uid: 6863 Time: 3.5
TraceDelay: RX 1012 bytes from 192.168.1.1 Sequence Number: 1 Uid: 6863
TXtime: +3500000000.0ns RXtime: +3514290866.0ns Delay: +14290866.0ns
TraceDelay TX 1024 bytes to 192.168.1.2 Uid: 7845 Time: 4
TraceDelay: RX 1012 bytes from 192.168.1.1 Sequence Number: 2 Uid: 7845
TXtime: +4000000000.0ns RXtime: +4019828717.0ns Delay: +19828717.0ns
POS: x=399.7, y=0
```

4.3.3 Третья версия модели

Третья версия модели представлена в файле wimaxv3.cc. В этой модели уже представлено большее количество узлов: узел мобильной станции, узел базовой станции, узел ASN и узел CSN (ssNode, bsNode, ASN_Node, CSN_Node соответсвенно).

Маршрутизация производиться при помощи класса ns3::Ipv4StaticRoutingHelper. Адрессация происходит по адрессу группы 224.30.10.81.

Результат выполнения программы:

```
TraceDelay TX 1024 bytes to 224.30.10.81 Uid: 5838 Time: 3
TraceDelay: RX 1012 bytes from 192.168.1.2 Sequence Number: 0 Uid: 5838
TXtime: +3000000000.0ns RXtime: +3014298353.0ns Delay: +14298353.0ns
TraceDelay TX 1024 bytes to 224.30.10.81 Uid: 6818 Time: 3.5
TraceDelay: RX 1012 bytes from 192.168.1.2 Sequence Number: 1 Uid: 6818
TXtime: +3500000000.0ns RXtime: +3509725446.0ns Delay: +9725446.0ns
TraceDelay TX 1024 bytes to 224.30.10.81 Uid: 7784 Time: 4
TraceDelay: RX 1012 bytes from 192.168.1.2 Sequence Number: 2 Uid: 7784
TXtime: +4000000000.0ns RXtime: +4016596592.0ns Delay: +16596592.0ns
TraceDelay TX 1024 bytes to 224.30.10.81 Uid: 9716 Time: 5
TraceDelay: RX 1012 bytes from 192.168.3.2 Sequence Number: 0 Uid: 9716
TXtime: +5000000000.0ns RXtime: +5015880229.0ns Delay: +15880229.0ns
TraceDelay TX 1024 bytes to 224.30.10.81 Uid: 10668 Time: 5.5
TraceDelay: RX 1012 bytes from 192.168.3.2 Sequence Number: 1 Uid: 10668
TXtime: +5500000000.0ns RXtime: +5511418431.0ns Delay: +11418431.0ns
TraceDelay TX 1024 bytes to 224.30.10.81 Uid: 11690 Time: 6
```

```
TraceDelay: RX 1012 bytes from 192.168.3.2 Sequence Number: 2 Uid: 11690
TXtime: +6000000000.0ns RXtime: +6006845525.0ns Delay: +6845525.0ns
POS: x=399.7, y=0
```

Литература

[1] Львовский С. М. *Набор и верстка в системе LaTeX.* М.: МЦНМО, 2006.

[2] Git - book. http://git-scm.com/book/ru.

[3] В. Вишневский С.Портной И.Шахнович. *Энциклопедия WiMAX путь к 4G.* Техносфера, Москва, 2009.

[4] ns-3 documentation. https://www.nsnam.org/docs/doxygen/index.html.

[5] Tex - Википедия. http://ru.wikipedia.org/wiki/TeX.

[6] Latex - Википедия. http://ru.wikipedia.org/wiki/LaTeX.

[7] Git - Википедия. http://ru.wikipedia.org/wiki/Git.

[8] Wimax - Википедия. http://ru.wikipedia.org/wiki/WiMAX.

[9] Wimax forum. http://www.wimaxforum.org/.

[10] Nsnam. https://www.nsnam.org/wiki/Main_Page.