前言

1. 网络模拟是指采用计算机软件对网络协议、网络拓扑、网络性能进行模拟分析的一种研究手段。
2. NS2采用OTcl和C++两种程序涉及语言进行开发设计，且软件结构相对松散，各个分析工具各自为政；文档资料纷繁复杂；对网络技术的原理理解不透彻且对网络原理和网络模拟的对照关系及其差异把握不好。
3. 全书分为两个层次：1－4 基础部分； 5－7 提高部分
4. 初识NS2
5. NS2是一种面向对象的网络模拟器，它本质上是一个离散事件模拟器，其本身有一个模拟时钟，所有的模拟都有离散事件驱动
6. NS2可以说是OTcl的脚本解释器，它包含模拟事件调度器、网络组件对象库等。事件调度器控制模拟的进程，在适当的时候激活事件队列中的事件并执行。网络组件模拟网络设备或节点的通信，它们交换特定的分组来模拟真实网络情况，并记录到Trace文件以供分析
7. C++语言编写事件调度器和数据通道上的基本网络组件：Tcl/OTcl编写模拟代码，对网络拓扑、节点、链路等各种部件和参数进行方便快速的配置。两者通过TclCL自动连接和映射。
8. NS2的第一个版本起源于1996年7月31器，目前在研究NS3
9. OPNET NS2 MATLAB SPW GloMoSim SSFNET 共计6中可以进行网络模拟的软件，其比较见第7页
10. SPW作为专业的DSP模拟设计开发工具，主要应用在电子设计、通信设计和芯片设计领域，对移动通信系统也提供了强大的支持，适合做底层开发和模拟。
11. Qual 按照网络的七层架构，采用模块化设计。能够进行批量模拟。
12. 在Linux下安装时，./install + vi .bashrc 两步走
13. NS2学习之道7步走：在需要了解NS2的深层原理时需要学会查阅手册中相关内容并对照源码进行了解。

第三章

1.编写一个Tcl脚本，需要以下几个步骤：

创建模拟器对象：

设置跟踪文件

创建网络拓扑结构 节点属性

设置代理和应用层并进行绑定

设置节点事件和时间的对应关系

Run 模拟

Gnuplot进行画图

2.整个模拟过程从创建一个模拟器对象开始，通过调用这个对象的各个过程实现

3.事件调度器其实就是一个事件队列，所有的事件按事件的触发事件排列。事件就是一个时间和处理函数句柄组成的结构

4.单播节点包括地址分类器和端口分类器，实际网络中的IP地址仅仅是网络中某台主机的标识，实际模拟时并不一定要将节点的地址设置为IP地址的形式。端口分类器负责把将分组传递给相应的Agent对象。实际中网络中的端口就是传输层协议服务访问点。

5.多播节点主要由多播分类器和复制器组成。多播分类器根据源地址和目的地址，将分组分类。

6.NS2的链路（Link）与TCP/IP网络分层结构中数据链路不同，它同时实现了物理层、数据链路层以及部分网络层功能。

7．在NS2中可以通过创建双向链路（duplex－link）模拟双向链路，通过单向链路（simplex-link）创建单向链路。queue-limit 用于限制链路的队列大小。

8.一条双向链路相当于创建了两条方向相反的单向链路

9.TTL通常表示分组在被丢弃前最多能经过的路由器的个数。当记数到0时，路由器决定丢弃该分组，并发送一个ICMP报文给最初的发送者。

10.NS2可以设置双向及单向链路的属性，主要包括：方向、颜色、标签和队列位置。

11.当创建某个特定的Agent时，该Agent的C++ Agent类根据需要可以在其构造函数中调用bind函数与OTcl Agent类的内部变量绑定。

12.通常所有的OTcl类内部变量的初始化操作是在文件ns-default.tcl中进行的。

13.Agent可以构建和销毁网络层的分组，是网络层分组的起点和终点，同时，代理还可以实现各种不同层的网络协议。

14.TCP协议采用滑动窗口机制实现流控。窗口大小表示在最近收到的确认号之后允许传送的数据长度。TCP协议有多个不同的版本，不同版本之间的区别主要体现在拥赛控制算法的不同。

15.代理用于不同层协议的实现，对于传输层的协议，分组的大小和发送时间通常由代表应用层的独立对象来控制。对于底层使用的Agent，分组的大小和发送时间通常由Agent自己控制。

16.在NS2中实现一个新协议，可以通过创建一个新的协议代理来实现，见5.3节。

17.在NS2中，根据业务流产生的概率模型的不同，实现了四种流量产生器。在真实的网络中，应用程序一般是通过特定的应用程序接口（API）来访问网络资源。这些”API”就是经常提到的”sockets”。

18.在模拟过程中，数据跟踪和采集的方法有多种，大体可分为两大类：一类是在模拟的执行过程中直接显示跟踪数据；另一类且更为常用的是将采集到的数据直接存放到一个文件中，以便后期的处理和分析。

19.在NS2中主要有两种不同的数据监视器对象。Trace Monitor

20.通过QueueMonitor对象可以跟踪并统计分组的到达、离开、丢弃等相关数据。还可以有选择地计算这些数据的平均值。还可以对每个流进行监测统计，主要由三个类来完成对每个流的数据的汇总和统计。

21.Nam是基于Tcl/Tk 的动画演示工具，用来把模拟的过程用可是化的方式呈现出来，了解封包传送是如何从源端送到接收端。

22.有固定接入点的无线网络：通常所说的移动通信系统（例如手机）都属于这一类，其共同点是所有移动终端的通信都必须通过固定接入点（例如基站）来完成。还有WLAN系统。

23.无固定接入点的无线网络，通常称为Ad hoc 网络。网络中的节点既是通信终端，同时也是转发数据包的路由器。

24.移动节点和基本节点的一个主要区别是两个移动节点之间不需要通过有线链路（Link）相连。移动节点是由一系列的网络构件构成的。

25.链路层的另外一个重要的功能就是将分组的目标站点的物理地址（MAC）地址写入数据分组的MAC头中。

26.Channel对象就能从一个网络接口对象收到分组，然后复制n份给同一信道上的其他n个网络接口对象。所有收到分组的节点根据自己的无线信号传输模型来判断是否能正确接收到分组。

27.利用threshold来计算接收功率阈值：（其中的源文件值得研究一下，对于无线传输模型研究有益处）

28.利用setdest来自动生成运动场景文件 ;cbrgen.tcl 来生成TCP流或者CBR流

29.物理信道的参数设置要在创建信道对象之前。

30.TCP数据段头部专门有个窗口大小字段用于窗口通告

31.慢启动门限一般代表我们对网络容量的一个评估。

32.慢启动和拥塞避免、快速重传和快速恢复算法 组成了TCP协议拥塞控制的核心。

33.为什么第一次发送数据包是两个，是由选择的TCP版本RFC793edu实现的

34.在实际TCP协议中，对一个成功接收的数据包的恢复包头中有个确认序号字段指向下一个可接收的TCP序列号。

35.决定局域网特性的主要技术要素为网络拓扑，传输介质与介质访问方法。

36.CSMA/CD 碰撞监测就是指计算机边发送数据边监测信道上的信号电压大小，当几个站同时发送数据时，总线上的信号电压摆动值将会增大（互相叠加）

37.当前局域网使用的以太网帧帧主要有四种：802.2 802.3 Ethernet II,SNAP.以太网帧的最小长度为64个字节，最大长度为1518个字节。

38.VLAN是为解决以太网的广播问题和安全性而提出的一种协议，它在以太网帧的基础上增加了VLAN头，用VLAN ID把用户划分为更小的工作组，限制不同工作组间的用户在网络协议的第二层进行互访（即传统的局域网互访）。虚拟局域网的好处是可以限制广播范围，并能够将不在同一物理网段上的计算机组成虚拟工作组，动态管理网络。

39.网络上存在两种信息的交换，一种是网络业务信息的交换，；另一种是路径信息的交换。路由。路由算法主要负责路径信息的交换和定期更新。

40.NS2实现了典型路由算法的模拟，如单路径和多路径、距离向量和链路状态算法以及平坦和分层路由算法等。但是NS2本身尚没有提供对现有网络协议的支持，可以从其他地方移植或者自行添加NS2对特定路由的支持

41.ns-BGP 2.0工程在NS-2.27 版本上完整地移植和实现了路由协议BGP-4,具体下载安装和使用方法可以参照其网站提供的文档。

42.一个移动节点是一个基本节点对象附加了无线功能模块后的节点，并且移动节点能在一个假设的拓扑中移动，在一个无线信道中接收和发送信号。

43.God对象存储了移动节点的总数和一个用于记录从一个节点到另一个节点的最短路径和条数的表格。采用移动模式文件主要是为了模拟脚本的清晰易读。

44.队列管理主要管理网络节点的缓存队列（Queues），决定何时接受或者丢弃一个包。队列调度机制位于队列管理之后，主要解决多个队列中的数据包谁先输出的问题。适当的选择队列管理和队列调度算法，可以为网络流量提供在带宽、时延、丢包率等方面的保证，可以提供网络资源的公平分配方案，并且将减少整个链路在数据传输和控制方面的开销。

45.RED使用平均队列长度度量网络的拥塞程度，然后以线性方式将拥塞信息反馈给端系统。

46.队列管理算法一直是一个比较热门的研究领域，各种新出现的基于智能算法的队列管理算法也正在不断涌现，可参见相关科技论文。

47.网络的拥塞控制机制就是要维持网络处于低延迟高吞吐量的状态，队列管理是网络拥塞控制和服务质量保证的基石。

48.各个数据源的窗口大小变化存在高度的同步性，即它们都在同一时刻进行丢包，丢包后同时调整发送窗口的大小。

49.如果想进一步比较两种队列管理算法的优劣，我们可以通过数据统计出采用两种不同队列管理算法时，网络数据包的排队时延，网络吞吐率，丢包率等参数来进行比较。

50.Qos可以控制各种网络应用和满足各种网络应用要求，可控制资源和支持裁剪服务。

51.Diifserv与IntServ不同，它不需要信令，即应用程序在发出报文前不需要通知路由器。各网络元件（路由器，交换机）提供报文分类、队列调度、流量整形等功能支持Qos

52.在差分服务中，网络的边缘设备对每个分组进行分类、标记区分服务DS域，用DS域来携带IP分组对服务的需求信息。

53.MPLS可以承载的报文通常是IP包，当然也可以改进直接承载以太包、ATM的AAL5包甚至ATM信元等。

54.在MPLS中，LSP建立之后，核心路由器只根据MPLS头标中的标记来区分分组所属的转发，不对IP头标进行解析。

55.MPLS技术的快速标签转发以及良好的可扩展性，特别是在流量工程方面的应用，使得MPLS技术成为下一代IP骨干网的关键技术。

56.从相同的源到相同的目的节点的所有数据流遵从同一通用策略，一个策略包含策略类型、目标率和其他策略参数。NS2中的策略表用于存储每个数据流的策略类型。

57.在边缘路由器，可以使用”addPolicyEntry”命令来添加或更新策略表。该命令包含表示边缘队列、数据流源和目的节点的边缘队列变量、策略类型、初始代码点以及它使用的其他参数。

58.在核心路由器，可以使用”addPHBEntry”来添加PHB表项，PHB表是核心路由器用来将代码点（CP）映射为物理和虚拟队列的一个映射表。

59.我们设置DiffServ策略就是要在核心节点新增PHB表项，在边缘节点新增路由策略表以对进入的网络数据流进行分类和标记。

60.构件的主要功能是在C＋＋中实现的，OTcl中的类则主要提供C++对象面向用户的接口。

61.TclCL机制使得OTcl类能直接调用C++类函数，OTcl和C++能够互相操作对方定义的数据，并使得C++类可以和OTcl类对应起来，OTcl类的分层结构与C++类的层次结构保持一致。当实例化一个构件对象时，NS2会同时创建一个OTcl对象和一个对应的C++对象，并且这两个对象可以相互操作。

62.NS2的框架的搭建都是基于TclCL的。TclCL机制封装了六个C++类:Tcl、TclObject、TclClass、TclCommand、EmbeddedTcl、InstVar. TclObject是所有C++类的基类。TclClass能为C++类和OTcl类建立关联和初始化TclObject类的成员方法。

63.NS2的构件库是一个层次结构，并且每个构件通常由相互关联的两个类来实现的，一个在C++中，一个在OTcl中。

64.每个C++类都有一个从TclClass类派生而来的静态类，该类的create函数接收OTcl基类传来的参数(类名)，创建相应的对象，并返回C++对象的指针。

65.当OTcl脚本执行 new时，通过init($self next) 调用SplitObject的init 构造函数，完成hash表中注册类，并调用create-shadow 调用hash表中对应静态类的class\_xxx的create函数，返回创建C++类的对象

new Agent-> Agent:init -> SplitObject:init -> SplitObject: create-shadow ->class\_agent:create

66.NS2是在C++对象的构造函数中使用bind函数来进行变量绑定。

67.当C++对象通过OTcl触发初始化时，command函数被登记到cmd{}过程中，为它们之间建立对应关系。OTcl对象通过cmd{}过程把参数传给对应C++对象的command函数，从而实现调用过程。

68.command函数根据OTcl传入的参数去调用相应的成员方法。

69.我们可以从源代码中借鉴很多经验，理解这些代码对构建自己协议是至关重要的，这也是精通NS2的必由之路。

70.在Scheduler中，run的调用意味着模拟开始，它不断地从事件队列中取出事件并调用dispatch

71. Classifier对象是节点的重要组成部分，它的功能是匹配一个分组，并基于匹配的结果把分组传递给相应的对象。每个Classifier对象都包括一个由slot number作索引的对象表，Classifier的工作是检查收到的分组slot number,然后把分组转发给由该slot number 索引的对象。

72. 定时器类通常是Agent类的友元。TcpAgent类中包含了一个定时器对象，TcpAgent类通过成员函数调用该定时器对象的对应方法。各种Agent都是以这种方式使用定时器的。要实现一种新的定时器，只需要继承抽象类TimerHandler并覆盖expire()函数即可。

73.当定义一个新分组类型时，应该把它添加到枚举类型packet\_t的所有枚举值的最后，同时这个分组类型的名字应该添加到p\_info类的构造函数中

74.Packet类是Event类的子类，这样分组就可以作为一个事件被调度器调度。Packet类的成员函数比较多，但都是围绕着三个方面定义的:一是分组空间的分配，二是分组数据的获取，三是分组头空间的分配与特定分组头地址的获取。

75.代理(Agent)可以构建和销毁网络层的分组，是网络层分组的起点和终点，同时，Agent还可以实现网络层、传输层等不同层的网络协议。Agent类通过为应用层程序提供API来控制分组的大小和发送时间的。

76.Agent类中有关分组构造的成员函数通常不会被派生类覆盖，而涉及发送和接收分组的成员函数通常需要被派生类覆盖。

77.应用层构建在传输层代理之上，Appliaction类是所有应用程序的基类，不能被实例化。它的派生类流量发生器和应用模拟器分别用在UDP代理之上和TCP之上。

78.TelnetApp有两种产生分组的方法。如果成员变量interval\_为非0，那么分组发送时间间隔符合均值为interval\_的指数分布;如果interval\_为0，那么分组发送时间间隔的选择按照tcplib经验分布。

79.在网络研究中，NS2离散时间网络模拟工具在研究协议实现、协议交互和大规模性能问题方面是一个强有力的工具。NS2的最大价值体现在网络研究中为研究者提供各种模拟网络环境，目前网络研究领域中的绝大部分实验都是在NS2中进行的。

80.本节我们研究的问题是，由于持续增加的尽最大努力投递的流不使用端到端拥塞控制机制，所导致的公平性问题和潜在的拥塞崩溃。

81.单单依靠终端节点使用端到端拥塞控制在尽最大努力链路上保证Internet的健壮性是不现实的，同样，也不可能依赖所有的开发人员把拥塞控制集成在他们自己的Internet应用里。网络必须参与其中来控制自己的资源利用情况。

82.利用路由器来支持端到端拥塞控制机制，并开发一种以路由机制的形式来限制带宽的激励机制。

83.进行网络协议的相关研究经常需要修改现有的协议代码，或者实现研究者提出的网络协议。在NS2中实现一个新的协议、必须要1.创建自己的分组格式 2.创建路由代理 3.创建路由表 4.代码集成

84.分组是用来在对象间传递信息的，要在节点之间发送或接收新的协议分组信息，就必须在Packet类的分组部分加上一个新的分组结构。一般每种路由协议都有自己的分组格式。

85.当定义一个新的路由协议时，首先要为该路由协议定义自己的分组格式。

86.定时器与调度器的区别与联系:

定时器TimerHandler(本身是Handler)定时(delay)产生事件Event(Packet属于Event)

Scheduler::instance().schedule(this, &event\_, delay);

而由于TimerHandler::handle(Event \*e) {expire(e)},即定时器的处理函数是expire(),