

NS

网络模拟 和 协议仿真

黄化吉 冯穗力 编著
秦丽姣 陆宏治



书中实例的源代码
可到人民邮电出版社网站下载

 人民邮电出版社
POSTS & TELECOM PRESS

NS

网络模拟
和
协议仿真

人民邮电出版社
北京

图书在版编目(CIP)数据

NS网络模拟和协议仿真 / 黄化吉等编著. — 北京 :
人民邮电出版社, 2010. 5
ISBN 978-7-115-22415-6

I. ①N… II. ①黄… III. ①计算机网络—计算机模
拟 IV. ①TP393

中国版本图书馆CIP数据核字(2010)第029385号

内 容 提 要

本书介绍了利用NS软件进行网络模拟和协议仿真的方法,内容既包括NS的安装、Tcl/Otcl语言、NS架构、仿真入门等基础知识,又涵盖了协议的移植、代码调试以及协议开发等有关知识。书中不仅给出了大量具有详尽注解、讨论NS基本功能的源代码程序,而且在人民邮电出版社网站(www.ptpress.com.cn)上提供了一个配套的无线Mesh网协议开发和仿真的完整实例以及本书的示例源代码。

本书适合利用NS软件从事网络研究、开发和性能模拟的工程技术人员阅读,也可作为网络仿真的培训教程,以及高等院校与网络课程相关的实验教程。

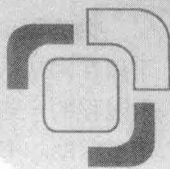
NS 网络模拟和协议仿真

-
- ◆ 编 著 黄化吉 冯穗力 秦丽姣 陆宏治
责任编辑 韦毅
 - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京铭成印刷有限公司印刷
 - ◆ 开本: 787×1092 1/16
印张: 16.75
字数: 433千字
印数: 1-4000册
- 2010年5月第1版
2010年5月北京第1次印刷

ISBN 978-7-115-22415-6

定价: 45.00元

读者服务热线: (010)67129264 印装质量热线: (010)67129223
反盗版热线: (010)67171154



以移动通信网络和互联网为代表的信息通信网络，已经成为人们日常生活和工作不可缺少的一部分。信息网络技术仍然在不断地完善和发展，大量的工程师和各类研究人员在这一领域工作。在进行网络的规划设计、网络协议的开发、网络路由交换与终端设备的研制等与网络技术研究有关的过程中，需要做大量的实验，由于存在各种条件的限制，不是所有的实验都能够通过物理实现的方法进行，于是基于计算机的网络模拟和协议仿真成为信息网络技术研究的重要手段。

目前已经有多种网络模拟的软件，其中 NS 以其源代码公开、软件模拟平台免费而获得广泛的应用，因为有大量的研究人员使用，人们比较容易获得针对许多特定应用的共享仿真软件。经过人们不断地补充完善，该软件已经成为学术界广泛用于网络技术研究和开发的模拟工具，所获得的结果有很好的仿真效果，同时也很容易进行交流。

然而，也正因为 NS 是一个非商业的软件，其用户界面不可能像商业软件那么友好，有关的帮助文件和介绍说明的资料也不尽完善，能够系统介绍和涵盖 NS 使用的各个方面的书籍相对也不多，因此要熟练掌握 NS，并将其作为工具进行网络技术的研究开发并不容易。除上面所说的原因之外，主要问题还在于要真正掌握 NS，需要有比较综合的知识：使用 NS 需要了解 Tcl 语言和 C++ 两种编程语言，特别是对 C++ 编程需要有较高的熟练程度；另外，NS 作为一个网络模拟的工具，要使用好它，还要对所模拟的网络的协议有较深入的理解和认识。这对于一个初学者来说并非易事。另外，即使对于有一定编程经验的读者，在学习 NS 的过程中，在缺乏指导的情况下，在进行新的协议开发的仿真过程中，也会遇到各种难以预测的困难。本书在编撰过程中，充分考虑了上述问题以及不同水平的读者的要求，本书有如下几方面的特点。

(1) 考虑到初学的读者对 NS 软件的入门的需要，本书从基本的 NS 平台安装、指令功能的介绍，到如何移植和开发网络协议，遵循由浅入深的原则，通过各种示例，全面细致地解析了 NS 的相关知识点，对书中所有出现的源代码，均给出尽可能详细的解析。

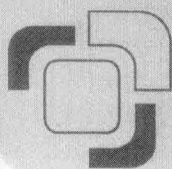
(2) 使用通俗易懂的语言和丰富的例子，深入地介绍 NS 的内部架构。通过具体的实例和详细的源代码分析，向读者详细介绍了 NS 中 OTcl 与 C++ 的关系，让读者对 NS 的影像对象机制有深刻的理解，帮助读者系统掌握 NS 中的层次关系。

(3) 重点介绍了常用的代码调试工具 GDB，通过几个调试实例详细分析了 NS 模拟仿真过程中可能出现的常见错误，以及出现这些错误的可能的原因，为读者在程序开发的调试阶段解决棘手的问题提供基本的方法。

(4) 本书在人民邮电出版社的网站上给出了与本书配套的在 NS 上开发基于简化 IEEE 802.16d Mesh 协议的具体实例，给出了实现其基本功能的、具有详细注释的完整源代码。在这个示例中融合了在 NS 上开发大型网络协议的流程和具体方法，帮助读者从较高的层次去理解和感受使用 NS 软件开发一套完整协议的过程。读者也可以在此基础上进行无线 Mesh 网协议的移植和开发。

为方便读者上机实践操作，本文中所有示例，包括基于 IEEE 802.16d Mesh 协议的实例程序，都可以在网站 www.ptpress.com.cn 上下载。本书中的很多内容是我们在学习和利用 NS 做网络协议开发过程的一些经验总结，希望无论是对于 NS 的初学者，还是对于有一定编程经验的研究人员，都有所帮助。书中存在的不足甚至错误在所难免，欢迎读者批评指正，并将意见和建议发送至本书责任编辑的信箱 weiyi@ptpress.com.cn。

作 者



第一篇 NS 的基本知识

第 1 章	绪论	
1.1	NS 的背景	3
1.2	网络模拟涉及的主要知识点	4
1.3	使用 NS 进行网络模拟的基本过程	5
	本章小结	6

第 2 章	NS 的安装和配置	
2.1	Linux 操作系统环境	7
2.1.1	Ubuntu 安装完成后的设置	7
2.1.2	Ubuntu 上 NS 必要工具和库文件的安装	8
2.2	NS 安装	8
2.2.1	环境变量的设置	8
2.2.2	安装结果的验证	9
2.2.3	NS 安装过程可能出现的错误分析	9
2.3	使用多个 NS 版本	10
	本章小结	11

第 3 章	Tcl/OTcl 语言简介	
3.1	Tcl	12
3.1.1	Tcl 编程环境	12
3.1.2	基本语法	13
3.1.3	字符串处理	14

3.1.4	数学运算	17
3.1.5	流程控制语句	19
3.1.6	列表	26
3.1.7	过程	27
3.1.8	数组	28
3.2	OTcl	30
3.2.1	类和对象	30
3.2.2	类的成员函数和成员变量	30
3.2.3	类的构造函数和析构函数	31
3.2.4	类的继承	31
	本章小结	33

第二篇 NS 的架构和网络模拟入门

第 4

章

NS 网络模拟入门

4.1	简单有线网络模型模拟	37
4.1.1	网络拓扑	37
4.1.2	模拟仿真脚本代码	38
4.1.3	仿真的结果分析	40
4.2	简单无线网络模型仿真	42
4.2.1	网络拓扑	42
4.2.2	模拟仿真脚本代码	42
4.2.3	仿真结果分析	45
4.3	有线+无线混合模拟	46
4.3.1	网络拓扑	46
4.3.2	模拟仿真脚本代码	47
4.3.3	仿真结果分析	50
	本章小结	51

第 5

章

NS 的架构

5.1	NS 的模拟实现机制	52
5.1.1	分裂对象模型	52
5.1.2	离散事件模拟	59
5.2	OTcl 与 C++的交互	59

5.3 NS 中的层次关系	66
5.3.1 层面对应关系	66
5.3.2 分组的流程	66
5.3.3 使用 NS 研究适合的层面	69
本章小结	69

第 6 章

章

NS 的基本组件和使用

6.1 节点	70
6.1.1 节点的结构	70
6.1.2 节点的配置	71
6.2 链路	73
6.2.1 单向链路	73
6.2.2 双向链路	74
6.3 NS 的调度器	74
6.4 NS 的定时器	78
6.5 协议的分组头	80
6.5.1 协议特定的分组头	81
6.5.2 添加新的分组头类型	82
6.5.3 在模拟中分组头的选择	83
6.5.4 与分组相关的类	84
6.6 NS 的链表 list	91
6.6.1 链表的源码分析	91
6.6.2 实际用例	93
6.7 Trace 文件格式	96
6.7.1 有线网络的 Trace 格式	96
6.7.2 无线网络的 Trace 旧格式	97
6.7.3 无线网络的 Trace 新格式	97
本章小结	99

第三篇 NS 仿真辅助及仿真结果分析工具

第 7 章

章

NS 仿真辅助工具

7.1 数据流生成工具 cbrgen	103
7.2 拓扑场景生成工具 setdest	104

7.3 简单动画显示工具 NAM	104
7.3.1 NAM 的使用	105
7.3.2 NAM 的界面	106
本章小结	106

第 8 章

gawk 语言简介

8.1 gawk 处理文档的过程	107
8.2 Pattern 和 Action	108
8.2.1 Pattern	108
8.2.2 Action	109
8.3 变量	109
8.3.1 内部变量	110
8.3.2 自定义变量	110
8.4 输出打印数据	111
8.4.1 print 函数	111
8.4.2 printf 函数	111
8.5 Action 中的流程控制语句	112
8.6 函数	114
8.6.1 内部函数	114
8.6.2 自定义函数	115
本章小结	115

第 9 章

使用 gnuplot 绘图

9.1 绘图环境和工作方式	116
9.2 plot 和 splot 命令	116
9.2.1 plot 绘制二维图形	117
9.2.2 splot 绘制三维图形	120
9.3 设置图形的显示属性	121
9.3.1 坐标轴	121
9.3.2 Label	122
9.3.3 图的大小和位置	123
9.4 综合技巧绘图实例	123
本章小结	128

第10章	Trace 文件分析	
10.1	性能参数分析模型	129
10.2	网络性能分析	131
10.2.1	OTcl 脚本代码	131
10.2.2	gawk 程序代码	135
10.2.3	执行分析	139
	本章小结	142

第四篇 NS 学习进阶

第11章	洪泛路由协议的移植	
11.1	协议移植的步骤	145
11.2	洪泛路由协议的原理	146
11.3	洪泛路由协议的移植	146
11.4	验证移植结果	149
	本章小结	155

第12章	无线节点通信范围的设置	
12.1	与通信范围相关的物理层参数	156
12.2	参数的计算	157
12.2.1	Threshold 的安装和使用	157
12.2.2	Txpower 的安装和使用	159
12.3	节点的参数设置	160
12.3.1	节点的默认参数	160
12.3.2	更改节点的通信范围	161
12.3.3	不同的节点不同的参数	162
12.3.4	验证结果	166
	本章小结	167

第 13 章 扩展 NS：业务流的流 ID 和 QoS 参数的添加方法

13.1 流量发生器的修改方法	176
13.1.1 修改 TrafficGenerator 类	177
13.1.2 修改流量发生器	178
13.1.3 增加新建参数的默认值	178
13.2 业务代理 (Agent) 修改	179
13.3 数据分组普通头部修改	179
13.4 打印 Trace 文件部分修改	180
13.5 修改结果验证	181
本章小结	185

第 14 章 流量发生器的原理和流量发生器的构建

14.1 流量发生器原理	186
14.1.1 流量发生器的数学模型	186
14.1.2 CBR 流量发生器模型代码分析	189
14.2 VBR 流量发生器	191
14.3 移植步骤	194
14.4 验证结果	194
本章小结	197

第 15 章 基于 NS 的新协议和算法开发

15.1 模块的体系架构	201
15.1.1 帧同步模块	202
15.1.2 主模块	202
15.1.3 调度树模块	204
15.1.4 消息管理模块	205
15.1.5 邻居管理模块	206
15.1.6 管理消息发送时序协调模块	207
15.1.7 数据调度器模块	208
15.2 接合部分代码	209
15.3 协议整合	212
15.4 场景仿真	212

15.5 仿真分析	219
15.5.1 节点的 ID 分配	219
15.5.2 调度树结构	220
15.5.3 邻居列表分析	220
15.5.4 节点对管理时序的竞争分析	220
本章小结	222

第五篇 代码调试

第 16 章

调试工具

16.1 GDB	225
16.1.1 关联 GDB 调试	225
16.1.2 GDB 常用调试命令	226
16.2 tcl-debug	229
16.2.1 tcl-debug 的安装	229
16.2.2 tcl-debug 的调试方式和调试命令	230
本章小结	230

第 17 章

NS 调试经验

17.1 常见错误分析	231
17.2 调试实例	232
本章小结	237

附录 A 802.16d Mesh 工作模式简介

A.1 无线 Mesh 网简介	238
A.2 IEEE 802.16 协议基础	239
A.2.1 协议栈参考模型	239
A.2.2 节点分类和数据调度方式	240
A.3 Mesh 模式的帧结构和 MAC 管理消息	241
A.3.1 Mesh 模式的帧结构	241
A.3.2 Mesh 模式下的 MAC 管理消息	242
A.4 节点接入网络过程	244
A.4.1 网络扫描, 请求入网	246

A.4.2	基本能力协商	246
A.4.3	节点认证	246
A.4.4	节点注册及获取节点 ID	246
A.4.5	IP、时间以及其余参数配置	246
A.4.6	接入完成	247
A.5	节点的邻域和扩展邻域	247
A.6	Mesh 模式管理消息发送时序的协调	247
A.6.1	物理邻居列表	248
A.6.2	Mesh 网络的同步	248
A.6.3	MSH-NENT 消息的调度	249
A.6.4	MSH-NCFG 消息和 MSH-DSCH 消息的调度	249
小结		252
附录 B	缩略语表	253
参考文献		256

第 1 章	绪论	3
第 2 章	NS 的安装和配置	7
第 3 章	Tcl/OTcl 语言简介	12

»»» 第一篇

NS 的基本知识

第一篇包含第 1~3 章的内容：第 1 章是绪论，主要向读者介绍 NS 的相关背景、网络模拟仿真的要点以及网络模拟仿真的基本过程；第 2 章简要介绍与 NS 的安装和配置相关的一些内容；第 3 章介绍 Tcl/OTcl 语言，这是用于编写仿真脚本的语言。

本书由浅而深地向读者逐步介绍 NS 模拟仿真的方法与应用的完整过程。本篇的内容是 NS 的基本知识，是学习 NS 模拟仿真前需要了解的基础。鉴于很多人在 NS 安装环节花了不少时间，建议读者能够细看第 2 章的内容，按书中的步骤去做可以节省一些时间；第 3 章的内容是日后编写模拟仿真脚本的基础，希望读者能够多花一些时间去掌握。

1. 介紹 編者
2. 研究 編者
3. 介紹 編者

第一編

第一章 緒論
一、研究之動機
二、研究之目的
三、研究之範圍
四、研究之方法
五、研究之經過
六、研究之結論
七、研究之貢獻
八、研究之不足
九、研究之展望
十、研究之附錄

网络协议的开发和完善需要进行许多验证和与性能相关的测试工作，在很多情况下这些工作是不可能都在实际的硬件系统上完成的，例如有一个算法需要几十个甚至上百个节点共同协作完成，受资源、经费、技术条件和场地等因素的限制，我们往往难以在实际的网络系统中完成验证和测试工作，这时就需要在虚拟的环境中进行模拟仿真。NS (Network Simulator) 是目前主流的网络模拟软件之一，其开源、免费和易于扩展的特性使其拥有庞大的用户群体。

1.1 NS 的背景

NS 是由 UC Berkeley 大学开发的，来源于 1989 年的 Real Network Simulator 项目，经过多年的发展，全世界各地研究者在此基础上不断地综合和完善，已经成为一个涉及网络各个方面的优秀的网络模拟工具。由于 NS 是一个免费的软件，源码公开，使用其研究网络协议也很方便，因此学术界使用 NS 研究和开发协议进行网络行为模拟和性能仿真的人员较多，每年在国内外发表的与网络技术相关的学术论文中，利用 NS 给出模拟结果的文章非常多，通过这种方法得出的研究结果也被学术界普遍认可。此外，NS 作为一种可用于辅助教学的工具，已被广泛应用在网络技术的教学方面。

NS 是一个面向对象的、离散时间驱动模拟器，使用 C++ 和 OTcl (面向对象的 Tcl 语言) 作为开发语言。选择这两种语言进行开发，主要基于如下考虑：①网络协议的模拟和实现需要高效地处理协议的报头信息，进行大量的数据处理，因此程序内部模块的运行速度是至关重要的，C++ 快速高效的特点非常适合于这部分的工作；②网络模拟围绕着网络组件、拓扑结构和其他环境参数而进行，一些参数需要快速改变以实现不同场景下的网络模拟仿真，此时网络环境的配置 (例如模拟时间、拓扑结构和节点的类型等) 就显得非常重要了，面向对象的 OTcl 脚本语言的灵活性就充分体现了其优势。两种语言的结合起到了取长补短的作用，使得 NS 能够充分体现其优势。

NS 已经经过了多年的发展，目前流行的是版本 2，也就是 NS2 (Network Simulator Version 2)，版本 2 中又有细分版本，例如 2.29、2.30 和 2.31 等。版本的每次更新都会做一些改动，可能是添加或删除了部分内容，也可能是更新了使用到的库文件。NS 的使用者往往是研究网络技术的某一层面，经常需要下载网络上已有的源码，然后进行移植，由于这些源码是在某个版本上开发的，可能在其他的版本上运行会有一些问题，这些问题对于一个经验丰富的 NS 使用者而言不会是太大的障碍。NS 使用者完全可以通过自己不断修改调试直至运行成功，获得所需要的网络仿真功能。但是对不熟悉 NS 的使用者来说，这是一项比较艰巨的工作，因此移植源码的时候需要进行版本匹配，以尽量减少错误。

尽管 NS 是一个非常适合于做研究的网络模拟软件，但是对于初学者而言，由于它不像商业的软件那样界面友好，从初学到上手，大多数的人通常需要半年左右的时间。有诸多原因使得 NS 的学习对许多人来说就像攀登陡峭的山崖，前进非常困难：NS 架构复杂（两种语言结合）、内容庞杂，使用环境是大多数人比较少接触的 Linux 系统，非可视化的界面，需要了解很多相关的知识和工具，参考资料很贫乏等。NS 本身只是一个网络模拟仿真的工具，学以致用，学好 NS 的最终目的是为了进行网络协议的研究和开发。因此，任何人都不希望仅在入门阶段就耗费过长的时间。经验的交流是降低学习 NS 门槛的最好方式，编写本书的目的就是希望通过我们的亲身体会，为 NS 学习者提供有益的帮助，加快其学习的进度，缩短其入门的时间。

NS 是在 Linux 平台上进行开发的，因此 NS 需要安装于 Linux 操作系统下。如果希望在 Windows 操作系统下使用 NS，可以先安装虚拟机，然后在虚拟机上安装 Linux 操作系统（这一方式对 CPU 和内存等硬件资源要求较高），或者安装模拟的 Linux 平台，例如 cygwin（cygwin 是一种 Windows 操作系统下模拟 Linux 的平台），然后在 cygwin 环境下安装 NS。

1.2 网络模拟涉及的主要知识点

用 NS 进行仿真，使用者需要了解掌握 3 个方面的知识：掌握网络模拟工具的使用，具有一定的 C++ 编程能力以及熟悉所研究的协议。

首先，使用者必须能够比较熟练地掌握网络模拟工具的使用。无论选用何种仿真工具：OPNET、OMNeT 或是 NS，都需要熟悉其使用方法。俗话说“工欲善其事，必先利其器”，工具是用于开发的，如果对工具尚不甚了解，当然难言利用其开展工作。当选用 NS 进行网络协议开发和仿真时，一个熟悉 NS 的人有可能几天或者十几天就可以移植并调试完成一个协议的源码，但对一个生手来说，这将是一个漫长的过程。

其次，使用者需要掌握一定的编程能力。作为商业的网络仿真软件，OPNET 和 OMNeT 都需要使用者具备一定的 C/C++ 编程能力，掌握 NS 的要求则更高。使用 NS 进行网络仿真需先分析仿真涉及的层次：基于 OTcl 编程的层次——若只利用 NS 自身已有的网络组件就可以完成仿真，则无须修改底层 C++ 代码，这时只需编写 OTcl 脚本；基于 C++ 和 OTcl 联合编程的层次——NS 自身没有使用者需要的网络组件或 NS 现有的网络组件不够完善，需要在底层编写 C++ 代码完成组件的添加或修改，然后再编写 OTcl 脚本仿真。大多数情况下的仿真任务（特别是利用 NS 进行协议的开发）都是后一种情形，所以对 C++ 做相对深入的了解是不可少的。

NS 只是我们研究过程中使用的工具，当我们进行网络协议的开发和仿真时，首先要对网络协议有充分的了解，如果我们对所研究的协议内容都尚不熟悉，当然就不可能知道如何设计软件的模块，或者不可能清楚可对 NS 现有的哪些协议进行改造利用，获得新的协议功能模块。那样的话，即使了解了工具也难以入手做网络协议的仿真和开发。

明白网络仿真需要掌握的知识点内容对我们是很有帮助的。有一些 NS 使用者，想移植他人的代码，可能会由于移植步骤有误而导致失败；有一些 NS 使用者想修改现有的一些协议，需要添加一些代码，语法错误会导致修改的失败；也有一些人对所需开发的协议没有完全明了，当然也难以成功。在使用 NS 时，出现问题往往还是综合性的，这实际上是很多人认为 NS 很难学的一个主要原因。只要我们清楚这些潜在的问题，有针对性地加以解决，在较短的时间内掌握 NS 的运用是完全