



国际知名大学原版教材

—— 信息技术学科与电气工程学科系列 20

Prentice
Hall

Distributed Systems
—— Principles and Paradigms

分布式系统 —— 原理与范例

Andrew S. Tanenbaum
Maarten van Steen



清华大学出版社

DISTRIBUTED SYSTEMS PRINCIPLES AND PARADIGMS

**ANDREW S. TANENBAUM
MAARTEN VAN STEEN**

*Vrije Universiteit
Amsterdam, The Netherlands*



Tsinghua University Press

(京)新登字 158 号

Distributed Systems—Principles and Paradigms

Copyright © 2002 by Prentice-Hall, Inc.

Original English Language Edition Published by Prentice-Hall.

For sales in Mainland China only.

本书影印版由培生教育出版集团授权清华大学出版社在中国境内（不包括香港、澳门特别行政区和台湾地区）独家出版、发行。

未经出版者书面许可，不得以任何方式复制或抄袭本书的任何部分。

北京市版权局著作权合同登记号：图字：01-2002-4447

本书封面贴有培生教育出版集团防伪标签，无标签者不得销售。

版权所有，翻印必究。

书 名：分布式系统——原理与范例

作 者：S. Tanenbaum

出 版 者：清华大学出版社（北京清华大学学研大厦，邮编 100084）

<http://www.tup.tsinghua.edu.cn>

印 刷 者：北京密云胶印厂

发 行 者：新华书店总店北京发行所

开 本：787×960 1/16 印张：52.75

版 次：2002 年 9 月第 1 版 2002 年 9 月第 1 次印刷

书 号：ISBN 7-302-05827-X/TP·3451

印 数：0001~5000

定 价：59.00 元

国际知名大学原版教材

——信息技术学科与电气工程学科系列

出版说明

郑大钟

清华大学信息科学与技术学院

当前,在我国的高等学校中,教学内容和课程体系的改革已经成为教学改革中的一个非常突出的问题,而为数不少的课程教材中普遍存在的“课程体系老化,内容落伍时代,本研层次不清”的现象又是其中的急需改变的一个重要方面。同时,随着科教兴国方针的贯彻落实,要求我们进一步转变观念扩大视野,使教学过程适应以信息技术为先导的技术革命和我国社会主义市场经济体制的需要,加快教学过程的国际化进程。在这方面,系统地研究和借鉴国外知名大学的相关教材,将会对推进我们的课程改革和推进我国大学教学的国际化进程,乃至对我们一些重点大学建设国际一流大学的努力,都将具有重要的借鉴推动作用。正是基于这种背景,我们决定在国内推出信息技术学科和电气工程学科国外知名大学原版系列教材。

本系列教材的组编将遵循如下的几点基本原则。(1)书目的范围限于信息技术学科和电气工程学科所属专业的技术基础课和主要的专业课。(2)教材的范围选自于具有较大影响且为国外知名大学所采用的教材。(3)教材属于在近5年内所出版的新书或新版书。(4)教材适合于作为我国大学相应课程的教材或主要教学参考书。(5)每本列选的教材都须经过国内相应领域的资深专家审看和推荐。(6)教材的形式直接以英文原版形式印刷出版。

本系列教材将按分期分批的方式组织出版。为了便于使用本系列教材的相关教师和学生从学科和教学的角度对其在体系和内容上的特点和特色有所了解,在每本教材中都附有我们所约请的相关领域资深教授撰写的影印版序言。此外,出于多样化的考虑,对于某些基本类型的课程,我们还同时列选了多于一本的不同体系、不同风格和不同层次的教材,以供不同要求和不同学时的同类课程的选用。

本系列教材的读者对象为信息技术学科和电气工程学科所属各专业的本科生,同时兼顾其他工程学科专业的本科生或研究生。本系列教材,既可采用作为相应课程的教材或教学参考书,也可提供作为工作于各个技术领域的工程师和技术人员的自学读物。

组编这套国外知名大学原版系列教材是一个尝试。不管是书目确定的合理性,教材选择的恰当性,还是评论看法的确切性,都有待于通过使用和实践来检验。感谢使用本系列教材的广大教师和学生的支持。期望广大读者提出意见和建议。

Distributed Systems—Principles and Paradigms

(分布式系统——原理与范例)

影印版序

本书的作者 Andrew S. Tanenbaum 早已为读者所熟悉,他撰写了多部关于计算机网络、操作系统的著作和教材。本书实际上是他在 1995 年出版的教材《分布式操作系统》(Distributed Operating Systems)的修订版。正如作者自己所指出的,从那时以来,技术的飞速发展和重大变化,例如,Internet 和 WWW 技术的发展(WWW 可以说是迄今最大的分布式系统),而在原来的那本书中甚至根本没有提到,因此,仅仅修订一下原教材,无论从范围还是内容已远远不能适应这种发展和变化,而是需要有一本新书,从更宽的视野和更广的意义上论述操作系统的一些问题,特别是分布式系统。这就是作者重新撰写原书并取名为《分布式系统——原理与范例》的原因。全书主要内容为 12 章,目录如下:

前言

第 1 章 总论

- 1.1 分布式系统定义
- 1.2 目标
- 1.3 硬件概念
- 1.4 软件概念
- 1.5 客户-服务器(C/S)模型
- 1.6 小结

第 2 章 通信

- 2.1 层次化协议
- 2.2 远程过程调用
- 2.3 远程对象调用
- 2.4 面向消息的通信
- 2.5 面向流的通信
- 2.6 小结

第 3 章 进程

- 3.1 线程
- 3.2 客户
- 3.3 服务器
- 3.4 代码迁移
- 3.5 软件代理
- 3.6 小结

第 4 章 命名

4.1 命名实体

4.2 定位移动实体

4.3 移去不再访问的实体

4.4 小结

第 5 章 同步

5.1 时钟同步

5.2 逻辑时钟

5.3 全局状态

5.4 选择算法

5.5 互折

5.6 分布式事务

5.7 小结

第 6 章 一致性与复制

6.1 引言

6.2 以数据为中心的一致性模型

6.3 以客户为中心的一致性模型

6.4 分布式协议

6.5 一致性协议

6.6 例

6.7 小结

第 7 章 容错

7.1 容错简介

7.2 进程恢复

7.3	可靠的客户-服务器通信	9.5	小结
7.4	可靠的组通信	第 10 章	分布式文件系统
7.5	分布式提交	10.1	SUN 网络文件系统
7.6	恢复	10.2	CODA 文件系统
7.7	小结	10.3	其他分布式文件系统
第 8 章	安全	10.4	各种文件系统的比较
8.1	安全简介	10.5	小结
8.2	安全通道	第 11 章	基于分布式文档的系统
8.3	访问控制	11.1	WWW
8.4	安全管理	11.2	Lotus Notes
8.5	例:KERBEROS	11.3	WWW 与 Lotus Notes 的比较
8.6	例:SESAME	11.4	小结
8.7	例:电子支付系统	第 12 章	基于分布式协调的系统
8.8	小结	12.1	协调模型简介
第 9 章	基于分布式对象的系统	12.2	TIB/RENDEZVOUS
9.1	CORBA	12.3	JINI
9.2	DCOM	12.4	TIB/RENDEZVOUS 与 JINI 比较
9.3	GLOBE	12.5	小结
9.4	CORBA, DCOM 以及 GLOBE 的比较	第 13 章	各章的阅读材料和参考文献

在结构上本书可分为“原理”和“范例”两大部分。第 1 章为总论,讨论了分布式系统定义、目标、硬件概念、软件概念、客户-服务器模型等内容。“原理”部分从第 2 章至第 8 章共 7 章组成,主要论述分布式系统中最为重要的一些基本概念和原理,包括通信、进程、命名、同步、一致性和复制、容错、安全性等;“范例”部分则由第 9 章至第 12 章共 4 章组成,分别介绍了分布式系统中的几个典型范例,由这些范例构成的几个主要系统,这些范例包括基于分布式对象的系统、分布式文件系统、基于分布式文档的系统、基于分布式协调的系统等。

本书是作者为计算机科学专业的高年级本科生或研究生的课程而写的,既系统地讲授了分布式系统的基本原理,技术的发展,也给出了一些范例和实际系统,每章都有问题与练习,供学生去深入思考,另外配有题解,是一本很好的教材。因此,我认为清华大学出版社以影印方式把该书引入国内,将会给计算机专业的高年级本科生和研究生提供一个很有参考价值的关于分布式系统的教材。对于从事分布式系统和分布式操作系统领域研究和开发的工作人员,本书也具有很大参考价值。

史美林 教授
清华大学计算机科学与技术系
2002 年 8 月

PREFACE

This book started out as a revision of *Distributed Operating Systems*, but it was soon apparent that so much had changed since 1995, that a mere revision would not do the job. A whole new book was needed. Accordingly, this new book has a new title: *Distributed Systems: Principles and Paradigms*. This change reflects a shift in emphasis. While we still look at some operating systems issues, the book now addresses distributed systems in a broader sense as well. For example, the World Wide Web, which is arguably the biggest distributed system ever built, was not even mentioned in the original book because it is not an operating system. In this book it rates almost an entire chapter.

The book is structured in two parts: principles and paradigms. The first chapter is a general introduction to the subject. Then come seven chapters on individual principles we consider most important: communication, processes, naming, synchronization, consistency and replication, fault tolerance, and security.

Actual distributed systems are usually organized around some paradigm, such as “everything is a file.” The next four chapters each deal with a different paradigm and describe several key systems that use that paradigm. The paradigms covered are object-based systems, distributed file systems, document-based systems, and coordination-based systems.

The last chapter contains an annotated bibliography, which can be used as a starting point for additional study of this subject, and the list of works cited in this book.

The book is intended for a senior-level or a graduate course in computer science. Consequently, it has a website with PowerPoint sheets and the figures used

in the book in various formats. The website can be located starting from www.prenhall.com/tanenbaum and clicking on the title of this book. A manual with solutions to the exercises is available to professors using the book in a course. They should contact their Prentice Hall representative for a copy. Of course, the book is also well-suited for individuals outside of a university setting wishing to learn more about this important topic.

A number of people have contributed to this book in various ways. We would especially like to thank Arno Bakker, Gerco Ballintijn, Brent Callaghan, Scott Cannon, Sandra Cornelissen, Mike Dahlin, Mark Darbyshire, Guy Eddon, Amr el Abbadi, Vincent Freeh, Chandana Gamage, Ben Gras, Bob Gray, Michael van Hartskamp, Philip Homburg, Andrew Kitchen, Ladislav Kohout, Bob Kutter, Jus-sipekka Leiwo, Leah McTaggart, Eli Messenger, Donald Miller, Shivakant Mishra, Jim Mooney, Matt Mutka, Rob Pike, Krithi Ramamritham, Shmuel Rotenstreich, Sol Shatz, Gurdip Singh, Aditya Shivram, Vladimir Sukonnik, Boleslaw Szymanski, Laurent Therond, and Leendert van Doorn for reading parts of the manuscript and offering useful comments.

Finally, we would like to thank our families. Suzanne has been through this process an even dozen times now. Not once has she said: “Enough is enough” although surely the thought has occurred to her. Thank you. Barbara and Marvin now have a much better idea of what professors do for a living and know the difference between a good textbook and a bad one. They are now an inspiration to me to try to produce more good ones than bad ones (AST).

Mariëlle knew what she was in for when I told her I was in the book-writing business again. She has been supportive from the start, noticing also that there was more fun and less frustration for me than the last time (“Are you writing chapters only once this time?”). Having Elke on your lap at 6 o’clock in the morning while writing is not such a good idea, but it kept me focussed on correctly setting priorities. In that respect, Max did a wonderful job as well, but being older than Elke, he also knew when it was better to play with someone else. They are great kids (MvS).

A GUIDE TO USING THIS BOOK

We have been using the material from this book for a number of years primarily for senior-level and graduate courses. However, it has also been used as the basis for one and two-day seminars on distributed systems and middleware for an audience consisting of (technical) ICT professionals. Below are some suggestions about how it can be used based on our experience.

Senior and Graduate Courses

For senior and graduate courses the material can typically be covered in 12-15 weeks. We have noticed that for most students, distributed systems appear to consist of a wealth of subjects that all seem to be tightly coupled to each other. The current organization of the book by which we present the subjects in terms of different principles and teach each principle separately has greatly helped in keeping students focused. The effect is that by the end of part one (Chaps. 1-8), before discussing paradigms, students already tend to have a fairly good impression of the overall picture.

Nevertheless, the field of distributed systems covers many different subjects, some of which are difficult to understand when studied for the first time. Therefore, we strongly encourage students to study the appropriate chapters as the course progresses. All sheets, which are available through the companion website (www.prenhall.com/tanenbaum), are handed out in advance allowing students to actively participate during class. This approach has been quite successful and is highly appreciated by the students.

All the material can be covered in a 15-week course. Most of the time is spent on teaching the principles of distributed systems, that is, the material covered in the first eight chapters. When discussing paradigms, it is our experience that only the essentials need to be presented. Details of each case study are more easily learned directly from the book than being taught during class. For example, we devote only a single week to object-based systems, despite the fact that there are approximately 80 pages devoted to these systems in the book. Below is a proposed class schedule showing which topics to cover in lectures.

Week	Topic	Chapter	Lecture on
1	Introduction	1	All
2	Communication	2	2.1–2.3
3	Communication	2	2.4–2.5
4	Processes	3	All
5	Naming	4	4.1–4.2
6	Naming	4	4.3
6	Synchronization	5	5.1–5.2
7	Synchronization	5	5.3–5.6
8	Consistency and replication	6	6.1–6.4
9	Consistency and replication	6	6.5–6.6
9	Fault tolerance	7	7.1–7.3
10	Fault tolerance	7	7.4–7.6
11	Security	8	8.1–8.2
12	Security	8	8.3–8.7
13	Object-based systems	9	All
14	File systems	10	All
15	Document-based systems	11	All
15	Coordination-based systems	12	All

Not all material is taught during class; students are expected to study specific parts by themselves, especially the details. When there are fewer than 15 weeks available for teaching, we suggest skipping the paradigm chapters and letting interested students read those on their own.

For junior-level courses, we recommend spreading the material over two semesters and adding lab assignments. For example, the students could work on a simple distributed system by asking them to modify components such that they can tolerate faults, handle multicast RPCs, and so on.

Professional Seminars for Industry

For one and two-day seminars we use the book as essential background material. Nevertheless, it is possible to cover the entire book in two days, provided that all details are skipped and an emphasis is put only on the essentials of distributed systems. In addition, to make the presentation more lively, it makes sense to reorder the chapters. The purpose of this is to show early on how the principles are used. Graduate students are used to getting 10 weeks of principles before seeing how they can be applied (if at all), but professionals are better motivated if they see how the principles are actually used. A tentative schedule for a two-day course that is divided into logical units is shown below.

Day 1				
Unit	Min.	Topic	Chap.	Emphasis
1	90	Introduction	1	Client/server architecture
2	60	Communication	2	RPC/RMI and messaging
3	60	Coordination-based systems	12	Messaging issues
4	60	Processes	3	Mobile code & agents
5	30	Naming	4	Location tracking
6	90	Object-based systems	9	CORBA

Day 2				
Unit	Min.	Topic	Chap.	Emphasis
1	90	Consistency and replication	6	Models and protocols
2	60	Document-based systems	11	Web caching/replication
3	60	Fault tolerance	7	Process groups and 2PC
4	90	Security	8	Basic ideas
5	60	Distributed file systems	10	NFS v3 and v4

Individual Study

The book can also be successfully used for individual study. If enough time and motivation is present, the reader is advised to go through the entire book cover to cover.

If there is not enough time to go over all the material, we suggest to concentrate only on the most important topics. The following table lists the sections we

believe cover the most important subjects relevant to distributed systems, along with illustrative examples.

Chapter	Topic	Sections
1	Introduction	1.1, 1.2, 1.4.3, 1.5
2	Communication	2.2, 2.3, 2.4
3	Processes	3.3, 3.4, 3.5
4	Naming	4.1, 4.2
5	Synchronization	5.2, 5.3, 5.6
6	Consistency and replication	6.1, 6.2.2, 6.2.5, 6.4, 6.5
7	Fault tolerance	7.1, 7.2.1, 7.2.2, 7.3, 7.4.1, 7.4.3, 7.5.1
8	Security	8.1, 8.2.1, 8.2.2, 8.3, 8.4
9	Object-based systems	9.1, 9.2, 9.4
10	Distributed file systems	10.1, 10.4
11	Document-based systems	11.1
12	Coordination-based systems	12.1, 12.2 or 12.3

It would be nice if we could make an estimate of how long it takes to cover the suggested material, but that depends so much on the background of the reader that it is impossible to say much in general. However, if this material is being read during the evening by someone with a full time job, it is likely to take several weeks at least.

CONTENTS

PREFACE

xvii

1 INTRODUCTION

1

- 1.1 DEFINITION OF A DISTRIBUTED SYSTEM 2
- 1.2 GOALS 4
 - 1.2.1 Connecting Users and Resources 4
 - 1.2.2 Transparency 5
 - 1.2.3 Openness 8
 - 1.2.4 Scalability 10
- 1.3 HARDWARE CONCEPTS 16
 - 1.3.1 Multiprocessors 17
 - 1.3.2 Homogeneous Multicomputer Systems 19
 - 1.3.3 Heterogeneous Multicomputer Systems 21
- 1.4 SOFTWARE CONCEPTS 22
 - 1.4.1 Distributed Operating Systems 22
 - 1.4.2 Network Operating Systems 33
 - 1.4.3 Middleware 36
- 1.5 THE CLIENT-SERVER MODEL 42
 - 1.5.1 Clients and Servers 42
 - 1.5.2 Application Layering 46
 - 1.5.3 Client-Server Architectures 50
- 1.6 SUMMARY 53

2 COMMUNICATION

57

- 2.1 LAYERED PROTOCOLS 58
 - 2.1.1 Lower-Level Protocols 61
 - 2.1.2 Transport Protocols 63
 - 2.1.3 Higher-Level Protocols 66

2.2	REMOTE PROCEDURE CALL	68
2.2.1	Basic RPC Operation	69
2.2.2	Parameter Passing	73
2.2.3	Extended RPC Models	77
2.2.4	Example: DCE RPC	80
2.3	REMOTE OBJECT INVOCATION	85
2.3.1	Distributed Objects	86
2.3.2	Binding a Client to an Object	88
2.3.3	Static versus Dynamic Remote Method Invocations	90
2.3.4	Parameter Passing	91
2.3.5	Example 1: DCE Remote Objects	93
2.3.6	Example 2: Java RMI	95
2.4	MESSAGE-ORIENTED COMMUNICATION	99
2.4.1	Persistence and Synchronicity in Communication	99
2.4.2	Message-Oriented Transient Communication	104
2.4.3	Message-Oriented Persistent Communication	108
2.4.4	Example: IBM MQSeries	115
2.5	STREAM-ORIENTED COMMUNICATION	119
2.5.1	Support for Continuous Media	120
2.5.2	Streams and Quality of Service	123
2.5.3	Stream Synchronization	127
2.6	SUMMARY	130

3 PROCESSES

135

3.1	THREADS	136
3.1.1	Introduction to Threads	136
3.1.2	Threads in Distributed Systems	141
3.2	CLIENTS	145
3.2.1	User Interfaces	145
3.2.2	Client-Side Software for Distribution Transparency	147
3.3	SERVERS	149
3.3.1	General Design Issues	149
3.3.2	Object Servers	152
3.4	CODE MIGRATION	158
3.4.1	Approaches to Code Migration	158
3.4.2	Migration and Local Resources	163
3.4.3	Migration in Heterogeneous Systems	165
3.4.4	Example: D'Agents	168

- 3.5 SOFTWARE AGENTS 173
 - 3.5.1 Software Agents in Distributed Systems 173
 - 3.5.2 Agent Technology 175
- 3.6 SUMMARY 178

4 NAMING

183

- 4.1 NAMING ENTITIES 184
 - 4.1.1 Names, Identifiers, and Addresses 184
 - 4.1.2 Name Resolution 189
 - 4.1.3 The Implementation of a Name Space 194
 - 4.1.4 Example: The Domain Name System 201
 - 4.1.5 Example: X.500 206
- 4.2 LOCATING MOBILE ENTITIES 210
 - 4.2.1 Naming versus Locating Entities 210
 - 4.2.2 Simple Solutions 212
 - 4.2.3 Home-Based Approaches 216
 - 4.2.4 Hierarchical Approaches 217
- 4.3 REMOVING UNREFERENCED ENTITIES 225
 - 4.3.1 The Problem of Unreferenced Objects 225
 - 4.3.2 Reference Counting 227
 - 4.3.3 Reference Listing 231
 - 4.3.4 Identifying Unreachable Entities 232
- 4.4 SUMMARY 238

5 SYNCHRONIZATION

241

- 5.1 CLOCK SYNCHRONIZATION 242
 - 5.1.1 Physical Clocks 243
 - 5.1.2 Clock Synchronization Algorithms 246
 - 5.1.3 Use of Synchronized Clocks 251
- 5.2 LOGICAL CLOCKS 252
 - 5.2.1 Lamport timestamps 252
 - 5.2.2 Vector timestamps 256
- 5.3 GLOBAL STATE 258
- 5.4 ELECTION ALGORITHMS 262
 - 5.4.1 The Bully Algorithm 262
 - 5.4.2 A Ring Algorithm 263
- 5.5 MUTUAL EXCLUSION 265
 - 5.5.1 A Centralized Algorithm 265
 - 5.5.2 A Distributed Algorithm 266
 - 5.5.3 A Token Ring Algorithm 269
 - 5.5.4 A Comparison of the Three Algorithms 270

5.6	DISTRIBUTED TRANSACTIONS	271
5.6.1	The Transaction Model	272
5.6.2	Classification of Transactions	275
5.6.3	Implementation	278
5.6.4	Concurrency Control	280
5.7	SUMMARY	288

6 CONSISTENCY AND REPLICATION

291

6.1	INTRODUCTION	292
6.1.1	Reasons for Replication	292
6.1.2	Object Replication	293
6.1.3	Replication as Scaling Technique	296
6.2	DATA-CENTRIC CONSISTENCY MODELS	297
6.2.1	Strict Consistency	298
6.2.2	Linearizability and Sequential Consistency	300
6.2.3	Causal Consistency	305
6.2.4	FIFO Consistency	306
6.2.5	Weak Consistency	308
6.2.6	Release Consistency	310
6.2.7	Entry Consistency	313
6.2.8	Summary of Consistency Models	315
6.3	CLIENT-CENTRIC CONSISTENCY MODELS	316
6.3.1	Eventual Consistency	317
6.3.2	Monotonic Reads	319
6.3.3	Monotonic Writes	320
6.3.4	Read Your Writes	322
6.3.5	Writes Follow Reads	323
6.3.6	Implementation	324
6.4	DISTRIBUTION PROTOCOLS	326
6.4.1	Replica Placement	326
6.4.2	Update Propagation	330
6.4.3	Epidemic Protocols	334
6.5	CONSISTENCY PROTOCOLS	337
6.5.1	Primary-Based Protocols	337
6.5.2	Replicated-Write Protocols	341
6.5.3	Cache-Coherence Protocols	345
6.6	EXAMPLES	346
6.6.1	Orca	347
6.6.2	Causally-Consistent Lazy Replication	352
6.7	SUMMARY	357

7 FAULT TOLERANCE 361

- 7.1 INTRODUCTION TO FAULT TOLERANCE 362
 - 7.1.1 Basic Concepts 362
 - 7.1.2 Failure Models 364
 - 7.1.3 Failure Masking by Redundancy 366
- 7.2 PROCESS RESILIENCE 368
 - 7.2.1 Design Issues 368
 - 7.2.2 Failure Masking and Replication 370
 - 7.2.3 Agreement in Faulty Systems 371
- 7.3 RELIABLE CLIENT-SERVER COMMUNICATION 375
 - 7.3.1 Point-to-Point Communication 375
 - 7.3.2 RPC Semantics in the Presence of Failures 375
- 7.4 RELIABLE GROUP COMMUNICATION 381
 - 7.4.1 Basic Reliable-Multicasting Schemes 381
 - 7.4.2 Scalability in Reliable Multicasting 383
 - 7.4.3 Atomic Multicast 386
- 7.5 DISTRIBUTED COMMIT 393
 - 7.5.1 Two-Phase Commit 393
 - 7.5.2 Three-Phase Commit 399
- 7.6 RECOVERY 401
 - 7.6.1 Introduction 401
 - 7.6.2 Checkpointing 404
 - 7.6.3 Message Logging 407
- 7.7 SUMMARY 410

8 SECURITY 413

- 8.1 INTRODUCTION TO SECURITY 414
 - 8.1.1 Security Threats, Policies, and Mechanisms 414
 - 8.1.2 Design Issues 420
 - 8.1.3 Cryptography 425
- 8.2 SECURE CHANNELS 432
 - 8.2.1 Authentication 433
 - 8.2.2 Message Integrity and Confidentiality 441
 - 8.2.3 Secure Group Communication 444
- 8.3 ACCESS CONTROL 447
 - 8.3.1 General Issues in Access Control 447
 - 8.3.2 Firewalls 451
 - 8.3.3 Secure Mobile Code 453