Addressing Threats and Security Issues in World Wide Web Technology[*][†]

Stefanos Gritzalis
Department of Informatics
University of Athens
TYPA Buildings, Athens GR-15771, GREECE
tel.: +30-1-7291885, fax: +30-1-7219561

Department of Informatics
Technological Educational Institute (T.E.I.) of Athens
Ag.Spiridonos St. Aegaleo GR-12210, GREECE
tel.: +30-1- 5910974, fax.: +30-1-5910975
email: sgritz@teia.ariadne-t.gr

Diomidis Spinellis
Department of Mathematics
University of the Aegean
Samos GR-83200, GREECE
tel.: +30-273-33919, fax: +30-273-35483
email:dspin@aegean.gr

May, 1997

Abstract

We outline the Web technologies and the related threats within the framework of a Web threat environment. We also examine the issue surrounding dowloadable executable content and present a number of security services that can be used for Web transactions categorised according to the Internet layering model.

[*] In *Proceedings CMS '97 3rd IFIP TC6/TC11 International joint working Conference on Communications and Multimedia Security*, pages 33–46. IFIP, Chapman & Hall, September 1997.

1 Introduction

The *World Wide Web* (WWW or "Web") is a distributed hypertext-based information system developed to be a pool of human knowledge allowing collaborators on remote sites to share ideas and information [BLCL+94]. The Web's hypertext and multimedia technologies make it easy for every user to roam, browse, and contribute. From a designer point of view the WWW is based on a client-server model. WWW is constructed from programs that make data available on the network. The WWW consists of:

- a set of servers, known as *Web servers*, which receive one request at a time and respond to that request without preserving state information, and

- a set of clients, known as *Web browsers*, which make requests based on user input and present the results.

Information in the WWW is accessed using a *Uniform Resource Locator* (URL) [BLMM94] which refers to any particular item and consists, in its most general form, of a request type, a host identifier, a port number, a user name, a password, and a pathname for the requested item. For example, http://www.aegean.gr:80/index.html indicates a hypertext transfer protocol access to the machine www.aegean.gr, on port 80, requesting the item named index.html.

Web documents are written in the *HyperText Markup Language* (HTML) [BLC95] which allows the specification of document structure, input fields, and, most importantly, hot spots containing hypertext links to other objects located on local or remote servers. Each link corresponds to a specific object on a specific Web server. A link is activated when a user clicks on its corresponding hot spot, which causes the browser tool to send a request to the Web server that stores the corresponding object. In turn, the Web server accepts the connection from the client and sends the requested data back, usually via a protocol known as *HyperText Transfer Protocol* (HTTP) [FGM+97]. In this way, WWW resources point to each other and present a world-wide information retrieval system.

Appart from the global WWW applications, companies are increasingly using WWW technologies to distribute and access corporate information using private Inernet-technology networks often referred to as *intranets*.

A server-technology protocol allows a client to activate a program on a server via a mechanism known as the *Common Gateway Interface* (CGI). Some servers may restrict access to certain objects or CGI scripts based upon the requesting user's identity, location, or other criteria; however, most servers on the Web do not currently restrict access to data in this manner. When the client receives the response, it may store the document in a local file. The client then uses content-type meta-information sent by the server to determine what application needs to be used to interpret and render the document.

The WWW is one of the most exciting and useful applications of the Internet. But, as is often the case, the designers of the WWW did not adequately consider protection and security when implementing the service; they opted for complete openness. As a result, the demand for security services for potential users has grown rapidly. Modern applications such as electronic commerce, business transactions, and information sharing have driven towards the development of many different approaches to provide security capabilities on the WWW.

The remainder of this paper is organised as follows: in Section 2 a brief description of threat agents and threats that exist in a typical Web environment are provided. In Section

3 security services that can support the protection process in distributed computer systems are discussed. Finally, Section 4 contains concluding remarks and our personal view of future research directions.

2 The WWW Threat Environment

2.1 Threat Agents in Web Technology

Security threats to computer systems fall into the following broad classes:

Leakage: the acquisition of information by unauthorised recipients

Tampering: the unauthorised alteration of information

Resource stealing: the unauthorised use of system facilities

Vandalism: interference with the proper operation of a system without gain to the perpetrator

When designing a secure open distributed system, the information that is used, in general, may include the following [FNS91]:

Ap the public key of a communicating entity A, which may be transferred to B

Bp the public key of B, which A may be received from a directory service

As the secret key of A, which must be hidden in A's end-system

Bs the secret key of B, which must be hidden in B's end-system

AMA information for achieving authentication A to B

AMB information for achieving authentication B to A

SSI secret session key information

AIa authentication information for users

PAu privilege attributes of users

PAa privilege attributes of applications

CAa control attributes of applications

The above information may be vulnerable to the following potential threats [FNS91]:

Undetected modification Information has been modified or originates from a false source. Modification can be performed by externals for the sole purpose of destruction or by internals trying to access resources for which she is not authorised. The following categories can be distinguished:

U1 Issued by false authority

U2 Modification by externals during communication

	U1	U2	U3	C	M1	M2	M3	M4
AP	x	x	-	-	-	-	-	-
BP	x	x	-	-	-	-	-	-
AS	-	-	x	x	-	-	-	-
BS	-	-	x	x	-	-	-	-
AMA	x	x	-	-	x	-	x	x
AMB	x	x	-	-	x	-	x	x
SSI	-	x	-	x	-	-	x	-
AIu	x	x	-	x	-	-	x	x
PAu	x	x	x	-	x	x	x	x
PAa	x	x	x	-	x	x	x	x
CAa	x	-	x	x	-	-	–	-

Table 1: Threats against security information

U3 Modification by internals.

Confidentiality violation C Secret information is read by someone it is not intended for.

Misuse Security information can be picked up by an eavesdropper for later masquerade or a user could try to use an out-dated key. The following categories can be distinguished:

M1 Use by correct initiator against wrong target

M2 Use by incorrect initiator against any target

M3 Replay attacks by externals

M4 Use of invalid information by internals.

Table 1 identifies the security information which is vulnerable to different threats.

In a typical WWW environment the user runs a Web browser application on a client residing on a multi-user machine, personal computer, or workstation running an operating system that may or may not provide security services. It is important to note that the lack of physical and/or software protection of the client operating environment severely limits the security-related guarantees that can be assumed. In many systems a *Web proxy* is used in a *Client-Server Interaction with Proxy* setup. The Web proxy forwards requests from the client to the server and passes responses from the server back to the client. It may also cache server data subject to a number of “freshness” criteria. Web proxies are commonly used to optimise performance by caching commonly accessed data, or to allow HTTP to pass through corporate firewalls.

As data flows through the aforementioned Web environment, in any instant, it can be in one of the following states:

Storage: the state where data is in either in volatile memory or in permanent storage, on either the client, an intermediate proxy, or the server computer system.

Processing: the state where operations are performed on data by the client, the proxy, or the server computer system.

Transmission: the state where data is transmitted between the actors of a Web system.

In the modern electronic commercial world, as the Internet accreted, the methods by which security violations can be perpetrated in open distributed systems (*modus operandi*, in criminology terms) depend upon obtaining access to existing communication channels or establishing channels that masquerade as connections to a principal with some desired authority. They include [CDK95]:

Eavesdropping: These attacks on a network can result in the obtaining of messages without authority. This may be achieved by obtaining messages directly from the network or by examining information that is inadequately protected in storage.

Message tampering: These attacks can be used to intercept messages and to alter their contents before passing them on to the intended recipient.

Masquerading: These attacks, also known as spoofing attacks, can be used to enable one party to masquerade as another party, without her authority. This may be done by obtaining and using another principal's identity and password or by using a token after the authorisation to use it has expired.

Replaying: These attacks are implemented by storing messages and, as a second step, sending them at a later date. The second step of this process may be done after authorisation to use a specific resource has been revoked.

2.2 Web Browsers and Downloadable Executable Content Risks

Recently, the development of downloadable executable content, using Webware technologies such as Java and ActiveX, has raised new risks [MF96].

While the advantages of using downloadable executable content come from the increase in flexibility provided by software programs and the wide access to existing software modules that may be located anywhere around the globe, it is this increase in flexibility and availability that may raise significant problems. For, instance, no user, when "surfing" the Web wishes applets or servlets that are executed within her browser to delete her files or even disclose private information over the network without the users consent.

Traditional applications, when running on a computer system, obtain access to certain resources of the system. In a similar way, downloadable executable content could also obtain access to such resources. While it is acceptable for traditional applications to utilise such resources, it is not desirable, at least to a certain degree, for downloadable executable content to do so. This is the case because downloadable executable content, i. e. the program that is running within a Web browser, is considered to be untrusted and as such could misuse a systems resources. For instance, a Java applet that runs within a Web browser should not be able to access vital for the system resources. If therefore, a Web browser that executes the Java code does not constrain the execution regarding the utilisation of the systems resources, severe security issues may arise.

An extreme solution to this problem would be to completely confine any downloadable executable content within the Web browser that is running it, hence not permitting any usage of the underlying systems resources. Nevertheless, such a solution is not a feasible one since one grants access to systems resources in order to make a program useful. For example, imagine a program, say an image processing tool, that requires to store its data, i. e. images, on the systems permanent storage device. Now assume that this program is downloadable executable content running in a Web browser that provides a completely sterile environment with no access whatsoever to systems resources. The program, in this context, becomes absolutely useless.

One, therefore, is required to strike the balance between totally confining a downloadable executable content within a Web browser and allowing it to freely handle system resources. By implication, one has to carefully consider what system resources and to what extent may be made available to a downloadable executable content, from within a Web browser, without endangering the systems security and at the same time guaranteeing the usefulness of the executable content.

3 Towards Secure WWW Transactions

3.1 System Security Services

In order to provide capabilities for protecting the system assets against the aforementioned threats system designers make use of specific security services. In the context *system security services* we include techniques as identification and authentication, access control, auditing, and encryption. The services are typically integrated into the Web servers and clients and often capitalise on the underlying operating system services.

Identification and Authentication

Identification and Authentication (I&A) is the twofold process of identifying an entity and afterwards validating the identity of this entity. In order to implement an authentication mechanism, one must determine what information will be used to validate the potential user. Whatever that information is, it can be described by one or more of the following categories:

- secret information (something the user *knows*)
- possession of a device (something the user *has*)
- biometrics (something the user *is*)
- location-based authentication (*somewhere* the user is) [DM96].

In most computer systems a user provides a user-id and a password which the O. S. verifies against an internally maintained database. In a network environment, it may be necessary to provide I&A information to multiple computer systems, requiring that the user re-enter authentication information multiple times. In [HKT96] a WWW authentication method is presented, which protects the authentication information and allows a user to only provide his authentication information once for an entire group of servers, residing possibly even in different Internet domains but belonging to a same logical group that has chosen to share its information among its members.

Currently many Web servers allow the identification of a user based on a (user-id, password) pair which is often transmitted in plaintext from the client to the server. Web servers are typically identified by their host name (e.g. www.microsoft.com) which offers only a weak identification guarantee given the security limitations of the Internet domain name system. More sophisticated schemes based on certificates validated through the use of certification authorities can be used to establish an identification trust model between Web clients and servers.

Access Control

A specific access control policy must be designed to dictate whether a given user can access a particular asset. The appropriate selected mechanisms aim to implement that specific policy. Within a host, the three most common access control mechanisms [Pfl96] are:

Mandatory Access Control (MAC): In these systems, the administrator assigns labels to user accounts and resources within the computer system. When a user makes a request to access a resource, the operating system (OS) retrieves the label associated with the authenticated user and compares it to the label associated with the resource. The OS then uses a predefined set of rules to determine whether the specific user with a particular label is permitted to access a resource with some, possibly different, label. Because most users have no way to change labels associated with resources, they cannot change who has access to a given resource, even if they control the contents of the resource itself.

Discretionary Access Control (DAC): DAC manages access to resources according to the identity of the user attempting the access. The OS compares the authenticated identity of the requester to the list of authorised users in the Access Control List (ACL) and allows or denies access accordingly. In DAC, each resource's ACL may be modified by the owner of the resource, regardless of who the owner is.

Role-based Access Control (RBAC): RBAC aims [SCFY94] [SJ97] to solve many of the management problems that arised with the DAC and MAC control types. Like in MAC all objects have a sensitivity label. But the subjects clearance is not statically assigned but is assigned on the basis of a request of a role by the subject. This specific request for a role by the subject is checked by a secure mechanism that works according to some predefined rules on trusted and verified data. If the request for a role is granted the subject receives its clearance.

Web servers typically couple I&A data with content or transactions that should be accessible to a set of users using control lists or ad-hoc protection schemes. In addition, the OS protection mechanisms are often used to protect the server and its content from unauthorised modification or disclosure.

Auditing

For many years, auditing controls and audit trails have been used to support backup procedures and security requirements in automated processing environments. Security auditing is defined as the process of collecting and recording security-relevant activities on a system. The audit records may be stored locally, or to a centrally located audit host. The protection mechanisms of the OS have to maintain the integrity of the audit log, so that once a security problem arises the event can be tracked to the perpetrator. Auditing is truly after-the-fact technique. Specifically, audit trails reinforce the use of security countermeasures by giving the security administrator a list of evidence to use in the prosecution of computer crime.

Most Web browsers keep a complete log of all transactions carried out. However, due to HTTP limitations coupled with privacy protection constraints the logs typically do not include the identity of the user who initiated a transaction, but only the name of the respective host. In addition, limitations of the IP protocol allow techniques such as *address spoofing* to be used in order to obscure even the host identity. However, given a higher level of assurance of user identities provided by secure I&A methods, the Web server audit records can provide a useful tool for satisfying security requirements.

Data Encryption

Given the insecure nature of public networks, in most environments data encryption services are used to provide the basis for the establishment of all services outline above. Encryption can be offered in two different forms, private and public key. The main advantage offered by public-key technology is increased security. Although slower than some private-key systems, public-key encryption generally is more suitable for modern applications, like electronic commerce, for it is more scalable to very large systems, it has a more flexible means of authentication, it can support digital signatures, and it enables non-repudiation procedures.

In the following sections we will describe concrete approaches of using public and private key encryption technologies for guarding against disclosure, fabrication, modification, and repudiation.

3.2 Solutions for secure WWW transactions

We will describe solutions for secure WWW transactions based on the Internet layering model [Com91, p. 146]. Based on this model the Web software and the associated network infrastructure is organised into four conceptual layers as follows:

External Applications Applications running on behalf of the Web client such as downloaded applets extending the client interface or the Web server such as CGI scripts providing dynamic content.

Web Applications The Web clients and servers communicating using the HTTP protocol.

Transport Layer The reliable end to end communication infrastructure over TCP.

Internet Layer The Internet Protocol (IP) routing and delivery layer.

In the following paragraphs we will describe the approaches used for securing transactions at each level. At the time of this writing a number of security proposals presented as IETF drafts (e.g. SEA, SKIP, Photuris, PCT) had reached their expiration date without being renewed or adopted and will therefore not be covered in the presentation.

Securing Web Transactions Using External Applications

Web transactions can be secured using specialised applications run on behalf of the server or the browser. Under this approach HTTP is used as a transport mechanism for transferring data that is then processed by the external application to provide services such as authentication and confidentiality. An example of this approach [WCS95] uses a browser API, the Common Client Interface, for communicating between the Web browser and an external application that uses PGP [Zim95] to encode the HTTP requests, decode the replies, and verify the signatures. This approach requires the least amount of modifications on the existing infrastructure, but can suffer from integration, performance, and usability problems.

Securing Web Transactions Using the Web Applications

A better integrated approach extends the HTTP protocol to deal with encryption and authentication. The Secure Hypertext Transfer Protocol (SHTTP) [RS97] specifies additions to the HTTP protocol to allow the negotiation of key management mechanisms, security policies, and cryptographic algorithms between parties for each transaction. It provides independently applicable security services for transaction confidentiality, authenticity, integrity and non-repudiation of origin. Under this approach Web clients and servers are extended to process the additional SHTTP headers and handle the requisite option negotiation. The extreme flexibility provided by the protocol makes its implementation, verification, and interoperation more difficult than other approaches. In addition the integration of security into the Web Application layer means that information that is transferred using other mechanisms (e.g. email) will need a separate security infrastructure.

Securing Web Transactions at the Transport Layer

Adding security extensions to a networking layer residing below the Web applications will automatically protect the applications that use the same transport layer service. Two proposals providing secure end to end TCP communication are the Secure Session Layer (SSL) [FKK96], and the Transport Layer Security [DA97]. The SSL protocol allows client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery. The TLS protocol is based on SSL 3.0 by standardising various technical details. SSL is integrated into a number of Web clients and servers and is also available in a reference implementation. Both protocols provide a robust and widely deployed security extension, but their integration at the transport layer may provide a less than perfect match for the implementation of transaction-oriented services such as electronic commerce.

One other approach at this layer is based on using a different model of client-server interaction eschewing the use of HTTP. DCE-Web [Sch97] uses secure RPC (based on Kerberos version 5) to handle client-server transactions. This approach can be better integrated with an operating system that already provides security services, but is difficult to deploy universally as at the time of this writing the requisite services were only provided within the context of the DCE.

Finally, it is important to note that an Application Programming Interface (API), the Generic Security Service API (GSS-API) has been developed [Lin97] to provide a consistent level of services and primitives to be used for ensuring authentication, integrity, and confidentiality.

Securing Web Transactions at the Internet Layer

Below the transport layer, the Internet Protocol can be enhanced in order to provide security services to all applications using it. Under this approach [Atk95c] two mechanisms are used for providing security services:

- the Authentication Header (AH) [Atk95a] provides authenticity and integrity by using a hash algorithm on the packet contents [MS95], and
- the IP Encapsulating Security Payload (ESP) [Atk95b] which uses DES to provide confidentiality [MKS95].

3.3 Protecting against downloadable executable content risks

As mentioned before, Webware is a very hot domain under continuous discussion and review. The Java programming language's fundamental security goal is to provide maximum protection for untrusted code using the *sandbox model* features. The sandbox restricts applet's actions in a dedicated area of the Web browser. Within its sandbox the applet may do anything it wants but cannot gain access to the user's file systems, network connections or other system resources.

The Web browser itself plays a large role in the security of the Java system. A Java enabled browser may include a Java interpreter and runtime library along with classes to implement a SecurityManager and various ClassLoaders. The security of Java thus relies upon the correct implementation of a fairly large code base. This situation is a result of Java's design choice to provide a very flexible model. Netscape Navigator is a Web browser based on the JDK. Netscape 2.x grants classes loaded from the local file system extra privileges (like JDK), but loads all applets via the class loader as if they were remote classes.

Microsoft's approach to executable content, ActiveX, uses a different approach. It is based on ActiveX authors signing their components using a key certified by a trusted third party. Under this approach the end-user can at least be reasonably confident about the origin of all downloaded content. Unfortunately, nothing in this model ensures that correctly signed malicious or simply erroneous components can not be properly signed and distributed.

3.4 Firewalls for increasing Web server's security

In theory, a firewall gives organisations a way to create a middle ground between networks that are completely isolated from external networks, such as the Internet, and those that are completely connected. An Internet firewall provides a simple way to control the amount and type of traffic that will pass between an organisation's internal and external network. It serves multiple purposes:

- it restricts people to entering at a carefully controlled point,
- it prevents attackers from getting close to your other defences, and
- it restricts people to leaving at a carefully controlled point.

A user can use a firewall in order to enhance their Web site's security in a number of ways. The most straightforward way use of a firewall is to create an "internal site", one that is accessible only to computers within the user's own network. If this is desirable, then the server has to be placed inside the firewall.

However, if the user wants to make the server available to the rest of the users, it must be placed somewhere outside the firewall. From the standpoint of security of an organisation as a whole, the safest place to put it is completely outside the LAN. This is known as a "sacrificial lamb" configuration. The Web server is at risk of being broken into, but at least when it is broken into it does not breach the security of the inner network. It is a bad idea to run the Web server on the firewall machine, for any security bug in the server will compromise the security of the entire organisation.

There are a number of variations on this basic setup, including architectures that use "inner" and "outer" servers to give the world access to public information while giving only to the internal network access to private documents.

4 Conclusions

The popularity of the WWW technology is due to several factors: its complete openness, the easy-to-use features of the modern browser tools, the potential for creating attractive presentations and links to other documents, locally or remotely, allowing integration of text, sound, and graphics. Security issues in the WWW technology are currently evolving rapidly. The reasons for this is the need for Web client/server hosts protection, for authentication of Web clients, servers, and users, for restricted access to Web assets, and for protecting confidentiality, authenticity, and integrity of Web data.

Many of the issues of Web security are still unresolved. Security problems are discovered on browsers and servers at an alarming rate, while important policy and technological questions have yet to be answered in a coherent and meaningful way. We believe that in the coming years the culmination and solidification of Web security research will transform the Web into a dependable, secure, and even more useful tool.

References

[Atk95a] R. Atkinson. IP authentication header. Request for Comments (Proposed Standard) RFC 1826, Internet Engineering Task Force, August 1995.

[Atk95b] R. Atkinson. IP encapsulating security payload (ESP). Request for Comments (Proposed Standard) RFC 1827, Internet Engineering Task Force, August 1995.

[Atk95c] R. Atkinson. Security architecture for the internet protocol. Request for Comments (Proposed Standard) RFC 1825, Internet Engineering Task Force, August 1995.

[BLC95] T. Berners-Lee and D. Connolly. Hypertext markup language - 2.0. Request for Comments (Proposed Standard) RFC 1866, Internet Engineering Task Force, November 1995.

[BLCL$^+$94] Tim Berners-Lee, Robert Cailliau, Ari Luotonen, Henrik Frystyk Nielsen, and Arthur Secret. The World-Wide-Web. *Communications of the ACM*, 37(8):76–82, August 1994.

[BLMM94] T. Berners-Lee, L. Masinter, and M. McCahill. Uniform resource locators (URL). Request for Comments (Proposed Standard) RFC 1738, Internet Engineering Task Force, December 1994.

[CDK95] George Coulouris, Jean Dollimore, and Tim Kindberg. *Distributed Systems: Concepts and Design*. Addison Wesley, 1995.

[Com91] Douglas E. Comer. *Internetworking with TCP/IP*, volume I: Principles, Protocols and Architecture. Prentice-Hall, second edition, 1991.

[DA97] Tim Dierks and Christopher Allen. The TLS protocol version 1.0. Internet Draft draft-ietf-tls-protocol-03.txt, Internet Engineering Task Force, May 1997.

[DM96] Dorothy Denning and Peter MacDoran. Location-based authentication. Computer Security Alert 154, Computer Security Institute, 1996.

[FGM$^+$97] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, and T. Berners-Lee. Hypertext transfer protocol – HTTP/1.1. Request for Comments (Proposed Standard) RFC 2068, Internet Engineering Task Force, January 1997.

[FKK96] Alan O. Freier, Philip Karlton, and Paul C. Kocher. The SSL protocol version 3.0. Internet Draft draft-ietf-tls-ssl-version3-00.txt, Internet Engineering Task Force, November 1996.

[FNS91] Claus Fritzner, Leif Nilsen, and Asmund Skomedal. Protecting security information in distributed systems. In *1991 IEEE Symposium on Security and Privacy*, pages 245–254. IEEE Computer Society Press, 1991.

[HKT96] Andrew Hutchison, Matthias Kaiserswerth, and Peter Trommler. Secure world wide web access to server groups. In *CMS '96 IFIP TC6/TC11 2nd joint working Conference on Communications and Multimedia Security*, pages 234–243. Chapman & Hall, 1996.

[Lin97] J. Linn. Generic security service application program interface, version 2. Request for Comments (Informational) RFC 2078, Internet Engineering Task Force, January 1997.

[MF96] Gary McGraw and Edward Felten. *Java Security Hostile Applets, Holes, and Antidotes*. J. Wiley & Sons, 1996.

[MKS95] P. Metzger, P. Karn, and W. Simpson. The ESP DES-CBC transform. Request for Comments (Proposed Standard) RFC 1829, Internet Engineering Task Force, August 1995.

[MS95] P. Metzger and W. Simpson. IP authentication using keyed MD5. Request for Comments (Proposed Standard) RFC 1828, Internet Engineering Task Force, August 1995.

[Pfl96] Charles Pfleeger. *Security in Computing*. Prentice-Hall, 1996.

[RS97] E. Rescorla and A. Schiffman. The secure hypertext transfer protocol. Internet Draft draft-ietf-wts-shttp-04.txt, Internet Engineering Task Force, March 1997.

[SCFY94] Ravi Sandhu, Edward Coyne, Hal Feinstein, and Charles Youman. Role-based access control: A multi-dimensional view. In *10th Annual Computer Security Applications Conference*, pages 54–62. IEEE Computer Society Press, 1994.

[Sch97] Brian Schimpf. Securing web access with dce. In *ISOC 1997 Symposium on Network and Distributed System Security*, pages 102–108. IEEE Computer Society Press, 1997.

[SJ97] Ben Smeets and Thomas Johansson. *Secure Storage and Retrieval in Medical Information Systems*. Lund University, 1997.

[WCS95] Judson D. Weeks, Adam Cain, and Briand Sanderson. CCI-based Web security. In *Fourth International World Wide Web Conference*, pages 381–395. O'Reilly & Associates, December 1995.

[Zim95] P. Zimmerman. *The Official PGP User's Guide*. MIT Press, 1995.

Biographies

Diomidis Spinellis holds an MEng in Software Engineering and a PhD in Computer Science both from Imperial College (University of London). He has provided consulting services to a number of Greek and international Information Technology companies and is a four times winner of the International Obfuscated C Code Contest. Currently he is lecturing at the University of the Aegean, Greece. His research interests include Software Engineering, Programming Languages, and Information Security.

Stefanos Gritzalis holds a BSc in Physics and an MSc in Electronic Automation both from the University of Athens, Greece. He is, also, pursuing a PhD degree on Distributed Systems Security, with the Department of Informatics of the University of Athens, Greece. Currently, he is an Assistant Professor with the Department of Informatics of the Technological Educational Institute (TEI) of Athens, Greece. His research interests include Distributed Systems, Computer Systems Security, and Operating Systems.