A Complete Strategy for Web Application Security

Hua Xu, Ronald J. Glotzbach, Nathan W. Hartman

Purdue University

Abstract

This paper is intent to develop a complete strategy to secure Web applications. The strategy is intended to improve the practices of the professionals associated with the development and operations of Web applications. Web application security is about protecting confidentiality, integrity, and availability of an organization's Web assets as well as the organization's reputation. The solution to Web application security is more than technology. It also involves policies, procedures, laws, people, and practices. Also, security is not a one-time effort. It should be an ongoing process integrated into the application development lifecycle. Security, like other Web application components, is best managed if planned at the initial phase of the application lifecycle. This strategy will help project managers and security professionals establish security policies, conduct risk assessment, and address potential risks in a cost-effective manner. It ensures system architects design secure application infrastructure. It makes sure application

developers write secure code. It helps security professionals conduct security review in a timely basis. It also enables system administrators to secure Web applications across the multiple layers of the application infrastructure. The ultimate goal of this strategy is to protect Web applications in a proactive, systematic, and holistic way.

1. Introduction

In general, security concerns confidentiality, integrity, and availability of systems and data. Confidentiality refers to the ability to ensure that information is private to the authorized parties and protected from unauthorized disclosure. Integrity reflects the accuracy of information and requires technology and processes that prevent unauthorized parties from inappropriately modifying information. Availability refers to the ability to ensure that information is accessible by its end users on a timely basis in order to meet mission requirements [14,15]. In the context of a Web application, security is about protecting confidentiality, integrity, and availability of an organization's Web assets (e.g. Web pages and customer databases) as well as the organization's reputation. Specifically, it is a process of making sure that, 1) data (e.g. system information and customer data) are private to the authorized parties when it is stored in the hosts or it is in transit, 2) data is protected from accidental or malicious modification when it is displayed on a Web site or transmitted over the Internet, and 3) Web site continues to function for legitimate users in order to meet mission requirements [12].

Web application security is not a one-time effort; it is a continuous risk management activity involving implementing technologies, policies and procedures, enforcing laws, and educating

and informing of the practices of the people associated with the development and operations of Web applications [9, 12]. Policies and laws help prevent criminal uses of security (i.e. Internet fraud) to some degree. Technologies, such as, encryption, the solution to protecting Internet communications, SSL, the protocol to protecting the channels of communication, and Intrusion Detection Systems (IDS), the tool to preventatively detect security breaches, help secure Web applications in a holistic way.

An understanding of Web application infrastructure will help address Web application security in a holistic approach [3, 12]. Web applications are built on multi-tiered architectures. A typical Web application is composed of a presentation logic tier, a middle tier (business tier and data access tier), and a data tier. They are hosted on the Web server, the application server, and the database server respectively. The presentation logic tier provides an interface for the end user into the application. If not properly authorized at the presentation logic tier, the user can access restricted areas or perform restricted operations on the Web server. The middle tier contains business rules and data manipulation, as well as a reusable interface to the database. If user activities and transactions are not audited and logged at this tier tier, it is hard for the system to detect and recover from security breaches. The data tier is where the data (i.e. customer information) resides. If customer data is not properly encrypted or hashed at this tier, sensitive information can be easily disclosed to and even modified by the unauthorized parties. In short, weaknesses at any tier leaves application vulnerable to attacks. Web applications should be secured across multiple tiers of the application architecture.

An understanding of application lifecycles will help address Web application security in a systematic way. The classic application lifecycle encompasses requirement gathering and analysis, design, construction, testing, deployment, and maintenance [13]. First of all, Web application security should start at the initial phase of the application lifecycle. It has been believed that adding a feature in a system after it has been designed costs ten times more than including this feature at the initial design phase [7], with no exception to security. Also, Web application is a dynamic system - any change in the development process will create new risk and make vulnerable what ever was secured [5]. This type of "build and fix" approach is expensive and dangerous. So, security should be a process integrated into the application lifecycle.

Furthermore, Web application security should be incorporated into the practices of each team member associated with the development and operations of Web applications. Project managers together with security professionals should establish security policies and conduct risk assessment at the initial design phase. System architects should design secure application infrastructure. Application developers should write secure code. Security professionals should review application security in a timely basis. System administrators should secure the application across the multiple layers, including networks, hosts, and application [12].

However, in many organizations, Web application security has not been built into the application lifecycle, addressed across the application infrastructure, or integrated into the practices of each team member.

2. Web Application Security Issues

Web applications have not yet been secured in a comprehensive way. Traditionally, Web application security has been treated as a network issue. Many organizations rely on firewalls and Intrusion Detection System (IDS) to protect their networks in order to achieve Web application security [5, 12]. However, Web applications are still faced with many security issues, ranging from the issues across the application infrastructure, to the lack of organizational policies and law enforcement. These security issues greatly reduce the confidentiality, integrity, and availability of Web applications.

2.1. Securities Issues Across Application Infrastructure

Web applications are composed of multiple layers, such as client browsers, hosts (e.g. Web server, application server, and database server), application or customer data stored on the hosts, and networks. Each layer has security issues and needs to be secured. The weakness in any layer makes Web application vulnerable.

> A vulnerability in a network will allow a malicious user to exploit a host or an application. A vulnerability in a host will allow a malicious user to exploit a network or an application. A vulnerability in an application will allow a malicious user to exploit a network or a host [12].

The network, as the infrastructure to support Web applications, has security vulnerabilities. First of all, network protocols (i.e. TCP/IP) are insecure in nature. Lacking the most basic security mechanisms (e.g. authentication or encryption), TCP/IP protocols are vulnerable to attacks such as eavesdropping, spoofing, and network layer denial of service. Eavesdropping is the act of monitoring network traffic for data (e.g. account credential and system configuration information) in plain text. By capturing administrator account, the hacker can gain privileged

access to the system [19]. Spoofing is the action of hiding true identifies on the network by using a fake destination address or hiding the original address of an attack. Spoofing threatens the integrity of the Web application by redirecting the user to a fake Web site. It also threatens authenticity of the application because the user is not able to tell who is the true sender of the message [9]. Network layer denial of service is the act of denying legitimate users access to Web server or Web service by flooding the network with traffic. It greatly reduces the availability of Web applications and affects the mission requirements of the organization [12]. Besides the insecure nature of protocol suites, poorly configured network devices, such as using default installation settings, exposing services that should be blocked, and leaving devices unpatched, also allow attackers to mount targeted attacks. In short, either the insecure nature of protocols or poor configuration management can reveal detailed information about network topology, system configuration, and network devices and leaves applications vulnerable to pointed attacks [12].

The host system of Web applications plays a critical role in Web application security. However, as a large and complex program, the host system may contain security holes. The following are the most common attacks on a host:

1. Malicious code, including viruses, Trojan horses, and worms, is one of the biggest threats to Web applications' integrity and continued operation. They may exploit security holes in the host, or gain unauthorized access to the host by underlying operating system [12, 17]. The cost caused by viruses, Trojan horses, and worms are expensive. According to Security Stat statistics, the worldwide impact of malicious code was 13.2 billion dollars in the year 2001 alone, with the largest contributors being SirCam at 1.15 billion dollars, Code Red (all variants) at 2.62 billion dollars, and NIMDA at 635 million dollars [18].

2. Hackers may gain unauthorized access by finding vulnerabilities in the host system [9, 17]. According to Computer Emergency Readiness Team (CERT) statistics, the number of reported computer security vulnerabilities almost doubled in 2002, with 2,437 in 2001 and 4129 in 2002, and dropped down slightly in 2003 with 3,784. However, the number of reported security incidents drastically increased with 52,658 in 2001, 82,094 in 2002 and 137,529 in 2003 [1].
3. Denial of service (DoS) attacks are the action of disrupting Web services by denying valid users to access the Web server, which greatly reduces Web application availability. Especially in an e-commerce system, DoS attacks are costly. Once an e-commerce system is shut down, customers cannot make purchases. In the meantime, the e-commerce site's reputation is greatly degraded [9].
4. The Web server may be used to distribute illegally copied software, attack tools, or pornography. This greatly destroys the organization's reputation [17].

Security has been traditionally treated as the issues of network layer and host layer. However, an estimated 70% of all security breaches today are due to vulnerabilities with the application itself[5]. HTTP protocol, like TCP/IP protocols, is not secure. Originally designed for hypertext transfer that didn't require persistent sessions, HTTP is now being used by applications with persistent sessions support. The inherently insecure nature of the HTTP protocol presents application developers more challenge, such as, secure authentication and session management mechanisms should be designed to track session state, data should be protected from parameter manipulation and information disclosure when it is transported over network [12]. In addition,

experimental data shows poor Web application design leads the majority of Web application security vulnerabilities. Malcolm Gin, a consult of Macromedia states,

> Most security vulnerabilities occur in areas of coding and design where secure design or implementation has lapsed. Sometimes the lapse comes from ill-considered design, sometimes it is the product of a rushed implementation, and sometimes it's the product of designer or developer arrogance. Regardless of how lapses occur, these vulnerabilities are consistently the most popular - and most devastating - openings for attacks[6].

2.2. Customer Privacy Issues

Organizations reassure their customers by publishing privacy policies on their Web sites. However, Marchany and Tront reported in 2002,

> The US Federal Trade Commission conducted a survey of 335 commercial Websites. Almost all the sites collected email address information from visitors but only 88% of the 335 sites had posted privacy policies. Twenty percent of these sites had policies that reflect the fair information principles of notice, choice and access security [11].

No customer wants the merchants to distributes sensitive information such as credit card number and SSN. However, US Bankcorp was sued in 1999 because it provided MemberWorks, a telemarketer, with sensitive information, such as customer bank account, credit card numbers, SSN, etc. [11].

2.3. Internet Fraud

Although technology provides a foundation for Web application security, technology is not the silver bullets in protecting Web applications. The lack of internal policies and law enforcement make the criminal uses of security much easier [9]. As a matter of the fact, instances of Internet fraud increased drastically in 2002 as compared to 2001. Losses reported by victims totaled 54 million dollars, versus 17 million dollars the year before, and complaints referred to law

enforcement totaled 48, 252, compared to 16, 755 in 2001. Auction fraud and non-delivery of merchandise were to top two reported crime, with Credit and debit card following them at 11.6%[4, 10].

3. A Strategy for Securing Web Applications

Web application security is technology centric, supported by organizational policies and procedures, law enforcement, and practices of the people associated with the development and operations of Web applications.

3.1. Technology Solutions

First of all, technology controls should be addressed across the application infrastructure, involving networks, hosts (Web server, application server, and database server), and application. Especially, the following technical controls should be used:

1. *Protecting Internet communications.* Protected communications are implemented through the use of data encryption methods and deployment of cryptographic technologies [14]. Encryption ensures the confidentiality of sensitive and critical information while it is in transmitted over the Internet. Public key cryptography with hash digest makes sure the integrity of information when it is in transit. By adding digital signatures, the authentication of the information and nonrepudiation can be ensured.
2. *Securing channels of communication.* Secure Sockets Layer (SSL) of TCP/IP is the most common form of securing communication channels. The SSL protocol provides data encryption, server authentication, and information integrity for TCP/IP connections [9].

Hence, it makes sure the confidentiality, authenticity, and integrity of information during the transaction between the merchants and customers.

3. *Intrusion Detection System (IDS) & anti-virus software.* IDS detects security breaches and initiates effective response if a security breach is detected [14]. IDS serves as a first line of defense against security breaches. Anti-virus software is installed on servers and user workstations, providing virus protection by identifying and eradicating the most common types of software viruses to ensure system and data integrity [14]. It is the easiest and least expensive way to prevent threats to system and data integrity.
4. *Authentication & authorization*. Authentication provides the means to verify the identity of the clients of Web applications and services. These clients might be end users, or other services (Meier et al., 2003). Authentication mechanism includes passwords, personal identification numbers, or PINs, and emerging technologies, such as token, smart card, and digital certificate [14]. *Authorization* specifies the allowed resources (e.g. files, databases, tables, and rows) and operations (i.e. on-line transactions) for the authenticated client [12].
5. *Access control enforcement*. Access controls define security policies and enforce the defined security policy on the authorized parties. The objective of access control is to preserve and protect data integrity and confidentiality. The common access control mechanisms include MAC sensitivity labels, DAC file permission sets, access control lists, roles, and user profiles [14].
6. *Auditing and logging.* The activities, such as successful and failed logon attempts, retrieval, modification, and deletion of data, network communication, and administrative functions, etc., should be audited and logged across the tiers of the application

infrastructure [12]. The auditing of security-relevant events and the monitoring and tracking of system abnormalities are key elements in the after-the fact detection of, and recovery form, security breaches [14].

7. *Input validation.* Unvalidated input has been continually listed as the top one Web application security flaw by OWASP (the Open Web Application Security Project) in 2003 and 2004. Unvalidated input may introduce buffer flows, cross-site scripting, and SQL injection and make application vulnerable in hence [16]. A good practice for Web application developers is to verify user input by constraining certain type, length, format, and range before it is used by the application and reject the invalid data as well.
8. *Session management.* Because the HTTP protocol does not provide the ability of persistent session, Web applications must create session by themselves. However, if the session tokens created by Web developers are not properly protected, an attacker can hijack an active session and assume the identity of a user. Web application developer should create a scheme to create strong session to protect them throughout their lifecycle.
9. *Exception management.* If error conditions that occur during normal operation are not handled properly, attackers can gain detailed system information, deny service, and cause security mechanism to fail or crash the server [16]. Web applications should avoid leaking information to the client in error message. Also, detailed error message should be logged.

3.2. Policies and Procedures

In order to minimize the risks, organizations should develop a coherent corporate policy that takes into account the Web assets that need to be protected, the nature of the risks, as well as the technologies and procedures required to address the risks [9].

First, customer privacy policies should be published on the Web site to reassure the customers. Customer privacy should includes the following information: 1) the explanation of how the organization defines privacy policies, 2) the choice of how the customer information is collected through the Web site, 3) how the customer data can be accessed by individual, and 4) assurances that customer data is securely stored by the organization [11].

Risk assessment, the first step of risk management, identifies the potential risks to system security and determines the probability of occurrence, the resulting impact, and additional safeguards that would mitigate this impact [14]. Risk mitigation takes place after risk assessment. It is the process of prioritizing, evaluating, and implementing the risk-reducing controls recommended by risk assessment process [14]. Based on the result of risk assessment, the recommended actions are prioritized, and evaluated according to the degree of feasibility and effectiveness. A list of possible controls is analyzed and then selected according to the impact of implementing and not implementing these controls. Finally, a safeguard action plan is generated which involves risk levels, prioritized actions, recommended controls, selected planned controls, responsible persons, start date, and target completion date.

Figure 1 shows the process to perform the above activities:

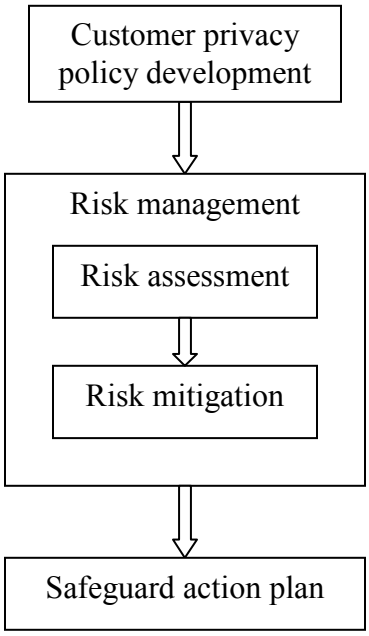

Figure 1: Web Application Risk Management Procedures

3.3. Law Enforcement

There are some control organizations or programs to enforce the computer crime laws. The Federal Trade Commission (FTC) takes the advantage of Internet for law enforcement, operating an online, real-time complaint form at its Web page. The database maintained by FTC contained 250, 000 consumer fraud by June 2000. The Internet Fraud Complaint Center, launched by the Federal Bureau of Investigation (FBI), also collects computer crime complaints from public at its Web site. Internet Fraud Council (IFC) is creating a clearinghouse of information regarding Internet fraud. It is also studying and quantifying these incidents and disseminating the information to law enforcement agencies [2]. These organizations and programs, have served a critical role for the United States by "facilitating the flow of information between law enforcement agencies and victims" [10].

4. Conclusions

Web application security is not a one-time effort; it should be an on-going process integrated into the application development lifecycle. In order to protect Web applications proactively, security should be taken into account at the initial phase of the application lifecycle. Technology is not the silver bullets of Web application security. In order to achieve Web application security in a cost effective manner, organizations should establish coherent corporate policies and procedures, taking into account the Web assets to be protected, the nature of the potential risks, and the technologies and processes required to minimize the risks, as well as the budgets. Also, government should enforce the laws to minimize the occurrence of criminal uses of security.

From the technology standpoint, comprehensive technical controls should be implemented across the application infrastructure, including networks, hosts, and application itself. Besides residing on secure networks and hosts, Web applications should be designed and developed with security in mind, including input validation, session management, and exception management. The basic security mechanisms, such as authentication, authorization, access control, and auditing and logging, should be implemented across the application infrastructure.

Bibliography

1.CERT/CC statistics 1988-2003(2004). Retrieved June 6, 2004, from http://www.cert.org/stats/cert_stats.html

2. Celentano, L., Edwards, W.C., & Farmer, J. J. (2000). *Computer Crime*. Retrieved Nov. 20, 2004, from http://csrc.nist.gov/publications/secpubs/computer.pdf

3. Chartier, R. (n.d.). *Application architecture: an n-tier approach*. Retrieved May 30, 2004, from http://www.15seconds.com/issue/011023.htm

4. Computer Security Spending Statistics (2004). Retrieved September 13, 2004, from http://www.securitystats.com/sspend.html

5. Complete Web application security. (2002). Retrieved June 6, from http://www.spidynamics.com/whitepapers/Webapp_Dev_Process.pdf

6. Gin, M. (n.d.). *How to design secure web applications*. Retrieved June 12, 2004, from http://www.macromedia.com/devnet/server_archive/articles/design_secure_webapps.html

7. Grance, T., Hash, J., & Stevens, M. (2003). Security Considerations in the Information System Development Life Cycle. *NIST SP 800-64*. Retrieved June 6, 2004, from http://csrc.nist.gov/publications/nistpubs/800-64/NIST-SP800-64.pdf

8. Kossakowski, K., & Allen, J. (2000). Secure Public Web Servers. *CMU/SEI-SIM-011*. Retrieved June 16, 2004, from http://www.cert.org/security-improvement/modules/m11.html

9. Laudon, K. C., & Traver, C. G. (2004). *E-commerce: Business. technology. society* (2nd). Addison-Wesley.

10. IFCC 2002 Internet Fraud Report (2002). Retrieved September 13, 2004, from http://www.ifccfbi.gov/strategy/2002_IFCCReport.pdf

11. Marchany, R.C., & Tront, J.G. (2002). *E-commerce security issues*. Retrieved September 13, 2004, from http://csdl.computer.org/comp/proceedings/hicss/2002/1435/07/14350193.pdf

12. Meier, J. D., Mackman, A., Dunner, M., Vasireddy, S., Escamilla, R., & Murukan, A. (2003). *Improving web application security: Threats and countermeasures roadmap*. Retrieved May 10, 2004, from http://msdn.microsoft.com/library/default.asp?url=/library/enus/dnnetsec/html/ThreatCounter.asp

13. Pressman, R. S. (2002). The process. In *Software engineering: A practitioner's approach* (pp.26-34) (5th ed.). New York: McGraw-Hill.

14. Stoneburner, G., Goguen, A., & Feringa, A. (2002). Risk Management Guide for Information Technology Systems. *NIST SP 800-30*. Retrieved May 16, 2004, from http://csrc.nist.gov/publications/nistpubs/800-30/NIST-SP800-30.pdf

15. Swanson, M. (2001). Security Self-Assessment Guide for Information Technology System. *NIST SP 800-26*. Retrieved May 22, 2004, from http://csrc.nist.gov/publications/nistpubs/800-26/sp800-26.pdf

16. The ten most critical web application security vulnerabilities. (2004). Retrieved June 6, 2004, from http://www.aspectsecurity.com/topten/topten_v2.pdf

17. Tracy, M., Jansen, W., & McLarnon, M. (2002). Guidelines on Securing Public Web Servers. *NIST SP 800-44*. Retrieved May 22, 2004, from http://csrc.nist.gov/publications/nistpubs/800-44/sp800-44.pdf

18. Web Defacement Statistics (2004). Retrieved September 13, 2004, from http://www.securitystats.com/webdeface.html

19. Wack, J. P., & Carnahan, L. J. (2001). Keeping Your Site Comfortably Secure: An Introducation to Internet Firewalls. *NIST SP 800-10.* Retrieved May 22, 2004, from http://csrc.nist.gov/publications/nistpubs/800-10/sp800-10.pdf

Biography

HUA XU is a Graduate Teaching Assistant in the Department of Computer Graphics Technology at Purdue University's West Lafayette campus. She received her B.S. in Computer Science from Wuhan University, PR China. She has a broad range of pursuits including software engineering, Web application development, as well as an interest in Web application security.

RONALD J. GLOTZBACH is an Assistant Professor in the Department of Computer Graphics Technology at Purdue University. He is also the Purdue football e-Stadium Application Manager for ITaP (Information Technology at Purdue). Ronald's research interests include leading-edge technologies that expand the boundaries of dynamic and interactive content delivered and collaborated on via the graphical communication tool that is the web.

NATHAN W. HARTMAN is an Assistant Professor in the Department of Computer Graphics Technology at Purdue University. Dr. Hartman received his Bachelor of Science and Master of Science from Purdue University and his doctorate from North Carolina State University. Dr. Hartman's primary teaching responsibility is undergraduate instruction in engineering graphics and graduate instruction in the foundations and philosophies of graphic science.