December 2001

E-Consulting on Web Server Security Infrastructure

Vijay Masurkar
Enterprise Services, Sun Microsystems, Inc.

E-CONSULTING ON WEB SERVER SECURITY INFRASTRUCTURE

Vijay Masurkar
Enterprise Services, Sun Microsystems, Inc.
vijay.masurkar@sun.com

Abstract

Web is ubiquitous, reaches more people and reaches them faster than most other services. As the use of Internet and the e-Commerce grew and the web became the most popular user interface for Internet applications, its security issues have become very visible. From the web infrastructure e-Consulting perspective, this paper generically examines the server-side security beginning with why and how it became an attractive target for attackers. From managing or recommending on electronic transactions to web authoring and firewalling, the pros and cons of the web server features are discussed with useful tips to web site and web server managers. Also, decisional tradeoffs faced by typical e-Commerce vendors in securing their web servers are presented.

Keywords: Web Server, Security, Encryption, Certificate, Firewall

Introduction

While availability, privacy, integrity and authenticity are usually the goals of any enterprise server security, interruption, interception, modification and masquerading may be considered the most common threats to those goals respectively. Interruption means the attacker's intention may be denial-of-service (DoS). In the case of interception, the attacker eavesdrops on information sent between client and server. Also, attacker may modify and replay information if interception is successful; as may be in the case of and interception of a credit-card transaction. Lastly, by masquerading, an attacker may pretend to be someone else. For example, a web site may represent a bogus organization.

Web has changed many of the industry's assumptions and traditions about computer security. In order to understand this, one must realize what web security means. There are many excellent references available [1][3][4][5][8][21] for understanding security and web issues in particular. The three major parts to web security involve securing the web server and the data it holds, securing the information that travels between the web server and the user and, the third, securing the user's computer where, typically, the web client browser resides. This paper focuses on the first part. The following sections will cover security issues in several key areas of the web server infrastructure e-Consulting; secure e-Commerce, web hosting, application service provider deployment etc. to name a few.

Background - The Evolution of Web Insecurity

How did the web insecurity infiltrate the Internet? Since the early days of the Internet, TCP/IP protocol suite has been known to have vulnerabilities as pointed by Bellovin (1989). Just a few years ago, the first generation of web servers made attractive targets for hackers because of the publicity that followed a break-in, commercial services that could be torn down or private information that could be stolen without much effort. In addition, the network access that the web servers extended to intranets made them the ideal gateways to sneak into a site.

In the first phase of the Internet, we used Internet as a transport mechanism. So, it automatically inherited all of Internet's vulnerabilities. Next was the advent of HTTP protocol and HTML, and, subsequently, the MOSAIC browser from the University of Illinois. Homogenous client & server design and the browser on the desktop turned the browser into an ubiquitous application.

Server-side scripts followed with HTTP & HTML and a 2-way communication between the server & client. Common Gateway Interface (CGI) scripts, search engines, VRML, XML and so on. HTTP requests a page, HTML page is delivered, form data returned and the CGI script on the server processed the form data. Subsequently, as the number of clients served by servers grew, it was the client-side scripts that reduced the load on the server. Servers sent bytecodes to clients for execution on the client side.

In the web communication, there are some basic assumptions. A web user assumes that the remote server is owned and operated by the organization that it seems to be owned by. The documents that the server returns are free from dangerous viruses and malicious intent. As for the webmaster, s/he keeps the faith that the user will not break into the web server or alter the contents of the web site. Further, the user will not try to gain access to documents that s/he is not supposed to or perform actions that will make the web server unavailable to others. And, lastly, the user is who s/he authenticates or claims to be.

On the other hand, there are assumptions on the user side as follows. The remote server will not record or distribute information that the user considers private, such as, for example, her web browsing habits. Both sides may assume that the connection is free from eavesdroppers and the information sent between browser and server is delivered intact. The web transactions are thus resting on these pillars of good faith. But, as we know, the real world wide web is quite different.

Know Thy Electronic Business Transaction

Netscape's Secure Socket Layer is a flexible, general-purpose encryption system which became very popular in the industry as e-Commerce took off. It operates at the TCP/IP transport layer, one level below the HTTP and provides confidentiality, authentication, integrity and nonrepudiation. SSL v3.0 was the first stable version with Diffie-Hellman key exchange and has been incorporated in all the popular commercial web servers. The Internet standards organization, Internet Engineering Task Force, reviewed SSL, proposed TLS (Transport Layer Sec Protocol), a derivative of SSL v3.0. Note that SSL (only) securely transmits credit card numbers; e.g. from the customer to the merchant. It doesn't check the validity of the number, authorization of the usage of the number or the transaction. That is why the server-side security is still open-ended inspite of the SSL usage.

SET (Secure Electronic Transaction) [6], for example, developed by Mastercard, Visa and some computer companies, is a higher level cryptographic protocol which provides an integrated system to handle the entire transaction including card authorization and final sale. In addition, since it cannot be used to encrypt arbitrary text messages, programs containing SET implementations with strong encryption have received export permission from the U.S. State Department.

There are many credit card systems in the market today used for e-Commerce. Any list anywhere would be insufficient to cover them all as new systems have been appearing in the market. So, it is more important to know how to evaluate them as a vendor on the server side and from the security point of view [20]. Typically, credit card numbers should not be stored on the web server considering that they represent consumer's private information. However, if they are stored for some compelling business reason, they should be stored encrypted and purged immediately after the transaction is completed. In addition, ability to process the card numbers by the server in real-time is necessary. Lastly, there are other issues such as related to export permission from the U.S. government (if you're consulting a U.S.-based vendor) for the system, charge-backs and giving credits to consumers, and how a server processes all of this protecting privacy of a consumer. An e-Consultant must analyze this in detail before making a selection for or recommending a credit card payment system for a server.

Privacy Issues Can Haunt You!

As people say, "browsing in a real shopping center is perhaps far more anonymous than browsing on the Internet!". The Internet is, of course, a microcosm of the debate over privacy and technology's impact on the collection of personal information. URLs you browse may be stored in multiple locations: your browser's history file, your organization's firewall, your ISP's firewall or proxy server, perhaps on the remote servers you visited, to name a few. As web server administrator or web site owner, it is your responsibility to make sure that you don't violate an individual's privacy or institutional or government regulations or laws.

There are many types of logs and identity information created on the server. Server Logs are typically turned on. Referer logs are sometimes generated using the "referer" field [23]. Proxies track events keeping proxy logs. On the Internet, cookies serve as the positive extension to the statelessness of the HTTP protocol, and browsers send cookies stored in their cookie file to their servers while accessing various sites [9].

What advice can you give to the web administrators who are your customers? While users have anonymizing proxies [10], remailers [10] to forge the source addresses and "cookie cutter" type of functions inside almost every browser to prevent or warn about setting cookies, the web site owners and administrators have a moral and professional responsibility to take certain actions. Draft and publish a privacy policy and keep it anonymous by default. One should know & be prepared for legal issues that one may have to face under unusual circumstances. It would not hurt to watch policy initiatives in the industry or sign up with TRUSTe organization [18]. Some important questions to think about are the following. Are server log files protected against a snooping user or an intruder? Do you use all the logged information you collect? If yes, for what reason/s? Be prepared to answer.

Is Your UNIX Server a Hacker's Paradise?

UNIX was designed in the heydays of the mainframe when user interactive computer was a novelty. Over the years, we added features that compromised security in exchange for enhanced user control. Irrespective of the complexity of the task the server is assigned, a few mundane tasks are absolutely necessary: applying vendor's opearting system patches, turning off unessential services, keeping minimum number of user accounts and setting directory and file permissions right.

Disable unnecessary web server features such as automatic generation of directory listing or server-side includes which are used to execute other programs or display data in the HTML files. Starting and stopping of the server should be performed without requiring root. Many times, with multiple system administrators, root access can become hazardous. Instead, consider running the web server in a change-root environment. Although resource intensive, it should be a routine to exercise to scan for [13], track and prevent denial-of-service (DoS) attacks. Further, limiting the processing resources is advised since DoS attacks attempt to extensively consume the computing resources.

UNIX operating system & Web Server logs provide plenty of supportive information to maintain a mission-critical system or to perform post-mortem examination in case of a security break-in. In addition to this, one may consider connection logging which records inbound connections to network services such as telnet (which is available by default in some, e.g. RedHat 6.2, but optional in some other UNIX systems,) and process accounting that shows all commands executed on the system. The extreme type of system instrumentation goes beyond mere logging alone. Examples are booby-trapped versions of standard utilities that set off alarms when an intruder starts rummaging around the system [25]. One entertaining episode utilizing such facilities is described in the attack on the Bell Laboratories research firewall [26].

In summary, below is a "top-10" checklist:

1. Have you installed all security-related patches? See CERT's [24] and vendor's security web pages.
2. Have you disabled all unnecessary services? Define unnecessary with respect to the business at hand.
3. Are there unnecessary accounts on the server? For example, "guest" or "test".
4. Do the Web Server's file permissions make sense? Tools such as COPS & TIGER [12] can assist here.
5. Is the Web Server running any unnecessary features - e.g. fancy CGI scripts, server plug-ins, etc.
6. Is the Web server running as root? Consider "chroot" and minimize administrators with root access.
7. Are the limits on the Web server usage conservative? Review with vendor's performance tools.
8. Do you monitor system and web logs for suspicious activity? Log analyzing tools are available[14].
9. Do you maintain tools to monitor file system integrity, - e.g. Tripwire [15]?
10. Do you back up your server? Have you made sure that backups can be restored successfully?

Looking at this from another point of view, being prepared as above, you have an opportunity to avoid losing control if a security breach happens. It is an opportunity one should not miss to control possible damage in future.

Benefits in the Complexities of Encryption & Digital Certificates

Encryption, in its basic form, has the characteristic of providing confidentiality, but digital certificates provide the additional security with authentication. Sensitive information on the server can be retrieved with a higher degree of confidence if certificates are exchanged to establish identities before the data communication begins.

The level of encryption (domestic or export grade) used or supported by the server is critical to a user's need. Currently, all commercial grade servers and browsers provide a full range of encryption as desired by the usage. Since public keys are public and could traverse an unprotected segment of network or domain, Certificate Authority (CA) for public keys came about. Private keys & certificate backups are web administrator's responsibilities. Also, web servers need to check of client certificates if clients are communicating using SSL and are presenting certificates. CA's private key protection is important since validity of the public keys of the clients relies on valid private key of the CA and server administrators must know how to protect the key. In addition, a web site that is adopting certificates need to assure that the CA is maintaining a sound certificate revocation process and keeping track of the validity of client certificates. And, lastly, servers must protect their private keys. So, there is an additional price to pay in terms of server administration for certificate level secure communication.

Avoiding Common Traps in Common Gateway Interface (CGI)

CGI scripts are small executable programs that are platform-independent way to reach an executable from a URL. They are independent of the main Web server binary. You can write CGI in many popular languages: Perl [16], Tcl, Java [22], Python, etc. Common risks with CGI involve unintentional leakage of information allowing intruders to break in or gain access to confidential documents. CGIs can be subverted into unauthorized modifications to files on the web site or server host and large programs, none other than the browsers themselves, can be tricked into executing destructive commands on the server host.

Misuse of interpreters as CGI Scripts have known to cause serious losses on the servers. Flawed memory management, passing unchecked user input to command interpreters, opening files based on unchecked user input, writing unchecked user input to disk are some other well-known examples of reckless usage of CGI [21].

The best CGI advice one can give is to use caution and log everything. Using temporary files and SUID/SGID scripts is typically unsafe. It is important to restrict access to sensitive CGI scripts, make use of system resource limits (e.g. "limit" or "ulimit" commands on the UNIX servers), rely on standard CGI libraries and avoid shell. In code development, perform regression and "Taint" checks which is an important security feature for Perl to verify if the program takes input from outside the program. Non-shell system calls avoid subprocess invocation of intermediate shell. And, lastly, increase the safety by making use of CGI Wrappers [17] [21] [23] which make SUID programs to execute with authors' permissions.

Controlling Access to Your Web Server

There are a variety of strategies that are employed today to control web server access. Three common ways are restricting access by using URLs that are hidden and unpublished, restricting access to a particular group of computers based on those computers' Internet addresses (also known as host-based addressing) and restricting users by their identity.

Hidden URLs are as safe as a key underneath your door mat. A well-known disclosure method of such information is via web "spiders", programs that sweep through all the pages of a web server. Security can be enhanced a little bit by using ".htaccess" files to employ password protection of your hidden page. If host-based addressing is based on Domain Name System (DNS), Internet Protocol (IP) spoofing is a possible attack method where the source address is forged to have an attacker's IP address. Hence, it is imperative to have a secure DNS server. The least effective way is to use an identity-based system to access a web site. The username/password based authentication is deployed in this case. Because passwords are easily shared or forgotten, many organizations recommend using a public certificate or a physical token such as a smart card. An obvious advantage of the identity-based system over a host-based one is that authorized users can access the web server from anywhere.

Treating Your Web-Authors

Web authors constitute a user community that you need to respect providing sufficient privileges if you are an administrator. Let us go over some important security considerations while e-Consulting. First of all, web authors, developers and administrators need separate categories of privileges. It may seem an overkill, but web authoring restrictions at the file system level makes sense when multiple authors work together and development happens in parallel. Multiple mechanisms for authors to update the web site have an uncanny way of allowing security to fall through the cracks. Do authors send their passwords for authentication or files in the clear? Think of sniffers, interception and password stealing. It is important to set up a methodology for source code control. Versioning of software can be an aid not only to diagnosing and solving the logistical problems but also to reconstruct software for a desired stage of development in the case of corruption. Lastly, check if your web developers work on scripts and modules on the "production" server. Obviously, this is a bad practice.

Firewall Them Off!

Firewalling a web server is a complex task. It starts with shopping for a firewall keeping in mind the key aspects. The choice of the operating system and consideration if it is hardened for access and misuse is one of the top criteria. Since administration is a necessary task, ease of administration becomes important. Review the protocols that are supported by the firewall engine; e.g. whether new, unusual protocols are supported. Many filltering capabilities are available in the market; - application, circuit -level proxies, packet filters, SMLI (Stateful Multi-Layer Inspection), etc. Each has pros and cons. Application proxies need customization per application. On the other hand, packet filters are faster than proxies. However, in general, SMLI provides significant speed and protocol state maintenance advantages over all the others.

Logging is necessary from the points of view of proactive monitoring and troubleshooting. Hence, exhaustive analysis/summary tools are desirable. Administrator's user interface needs to be friendly and intuitive with remote administration ability a required feature. Some vendors may allow source code inspection and one can take advantage of that. Check out the kinds of tunneling features that are supported by the firewall, - encrypting tunnel, virtual private networking (VPN), etc. Above all, simplicity is of paramount importance, since, as a general rule, complex software is a threat to security.

Many different configurations are possible offering pros and cons. If you're in e-Consulting, it is advisable to keep up with the firewall discussions in the industry to stay on top [7]. The two most fundamental considerations are how your organization safely allows internal users to browse the web and how your organization's public web site is made available to the rest of the world. In designing or troubleshooting a web server and firewall configuration for a customer, the web administrators/consultants must keep these primary questions in mind:

1. Where would be/is the customer's public and internal Web Server (if any) in relation to the firewall?
2. How would/do employees at your customer's site browse through the firewall?
3. How would/do you update the public web server through the firewall?
4. Can you, using the firewall tools, logs, data, proactively monitor the web server for compromise?
5. How does the firewall detect/block harmful data in the URL, in the request/ response, from inside/outside?

The placement of a web server with respect to the company's boundary firewall has been a subject of much debate. If a web server is outside the firewall, it cannot take advantage of the firewall's protection. If it is inside, users on the intranets can target it. In addition, a vulnerability in the web server can give an intruder a foothold behind the firewall. In the third alternative, the web server is on the firewall's third interface or what is called a DMZ ("de-militarized zone") whereby the internal or external users have to pass through the firewall rules to reach the web server. Typically, in the e-Commerce situation, the web servers communicate with database servers inside. A web site presents HTML forms, collects user responses into transaction records and then posts them into the database. Having a web server outside may present a serious risk of transactions being altered before reaching the internal database. So, a DMZ placement might be more secure in some cases. In summary, there are tradeoffs that must be evaluated for each business case.

Conclusions

It pays to proactively manage the web server security. The critically known issues, in particular, can be learnt quickly from organizations such as CERT [24]. Wherever possible, regarding the operating system platform issues in e-Consulting, utilize the vendor's recommendations for selecting, configuring, optimizing and tuning resources [19]. As we have seen, there are tradeoffs to be considered for every aspect of the web server security. What is presented here is a short set of practical guidelines. Designing, deploying and maintaining commercial web servers requires thorough analysis along these lines of all the alternatives at hand against the goals of the business.

References

[1] Security-standards activity by W3C & others: http://www.w3.org/Security/

[2] Bellovin, S., "Security Problems in the TCP/IP Protocol Suite," *Computer Communication Review*, Vol. 19, 2, pp. 32-48, April 1989.

[3] RSA Laboratories' FAQ: http://www.rsasecurity.com/rsalabs/faq/index.html

[4] Garfinkel, S., and Spafford, G, *UNIX & Internet Security*, O'Reilly & Associates, Inc., 1999.

[5] Java Applets - Sun Microsystems, Inc.'s documentation and white papers: http://java.sun.com/sfaq
[6] Secure Electronic Transaction (SET): http://www.mastercard.com and http://www.visa.com
[7] Firewall mailing list archive: http://www.greatcircle.com/firewalls
[8] Browser Security Pages & Alerts: http://home.netscape.com/info/security-doc.html
[9] Cookies - Netscape Cookie Specification: http://cgi.netscape.com/newsref/std/cookie_spec.html
[10] The Anonymizer: http://www.anonymizer.com/; Remailers: http://www.stack.nl/~galactus/remailers/
[11] Electronic Privacy Resources - Center for Democracy & Privacy: http://www.cdt.org/
[12] System Configuration Tools: COPS - ftp://ftp.cert.org/pub/tools/cops and TAMU - ftp://net.tamu.edu/pub/security/TAMU
[13] ISS Scanner (freeware) -ftp://coast.cs.purdue.edu/pub/tools/unix/scanners/iss
[14] Log Analyzers - http://www.uu.se/Software/Analyzers/
[15] Integrity Checker - ftp://coast.cs.purdue.edu/pub/COAST/Tripwire
[16] Perl documentation: http://www.perl.com
[17] CGI Wrapper site: http://www.umr.edu/ ~cgiwrap/
[18] TRUSTe organization: http://www.truste.org/
[19] Sun Microsystems Blueprints Home Page: http://www.sun.com/blueprints
[20] Garfinkel, S., and Spafford, G, *Web Security & Commerce*, O'Reilly & Associates, Inc., 1997.
[21] Rubin, A., Geer, D., and Ranum, M., *Web Security Sourcebook*, Wiley Computer Publishing, 1997.
[22] Arnold, K., and Gosling, J., *The Java Programming Language - Second Edition*, Addison-Wesley, 1997.
[23] Stein, L., *Web Security: A Step-by-Step Reference Guide*, Addison Wesley, 1998.
[24] Computer Emergency Response Team (CERT) at CMU: http://www.cert.org/
[25] Farmer, D., and Venema, W., "Being Prepared for Intrusion," *Doctor Dobb's Journal*, April 2001.
[26]Cheswick, W., and Bellovin, S., "An Evening with Berferd," *Proceedings of the Winter 1992 Usenix Conference*, 1992.