Security at the next level: are your web applications vulnerable?

White paper

Table of contents

Introduction 3
Why aren't web environments secure? 4
Hackers are one step ahead of your enterprise. 4
Passwords are not enough. 5
SSL and data encryption are not enough. 5
Firewalls are not enough. 5
Standard scanning programs are not enough. 5
A chain is only as strong as its weakest link. 5
It's in the code. 5
Manipulating a web application is simple. 5
How do you protect your site? 6
What do you need to do? 6
Anatomy of a web application attack 7
Attack techniques 7
Common application attack methodologies 8
Static vulnerability attacks 8
Dynamic vulnerability attacks 8
About HP WebInspect 11
Conclusion 11
Business case for application security 11

Application security is the trend of the future.

The need for security began with desktop computing when the only means of compromising data was by inserting a contaminated floppy disk into a PC. That was the anti-virus era. The need for security evolved with the Internet as more companies developed internal and external networks. That was the network security era. Now as companies leverage the power of the web, information security has evolved yet again: We are in the application security era.

Introduction

Web applications can take many forms—an informational website, an e-commerce site, an extranet, an intranet, an exchange, a search engine, a transaction engine, an e-business. All of these applications link to computer systems that contain weaknesses that can pose risks to your organization. Weaknesses exist in system architecture, system configuration, application design, implementation configuration and operations. The risks include the possibility of incorrect calculations, damaged hardware and software, data accessed by unauthorized users, data theft or loss, misuse of systems and disrupted business operations.

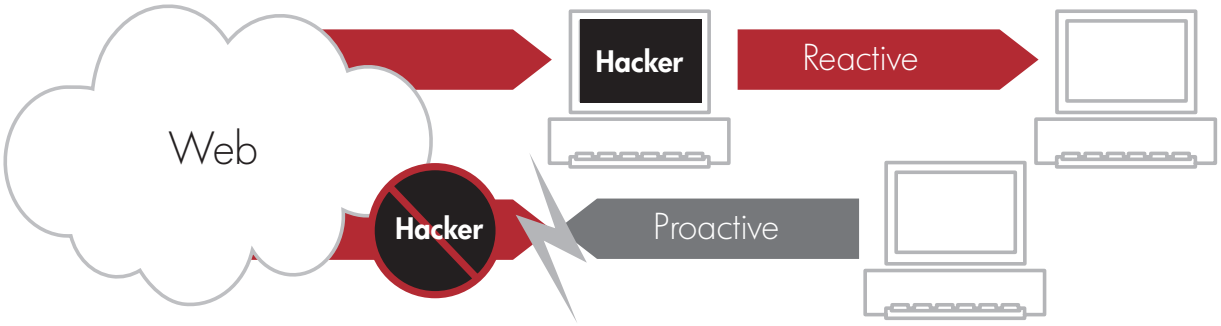
As the digital enterprise embraces the benefits of e-business, the use of web-based technology continues to grow. Most organizations today use the web as a way to manage their customer relationships, enhance their supply chain operations, expand into new markets and deploy new products and services to customers and employees. However, successfully implementing the powerful benefits of web-based technologies cannot be achieved without a consistent approach to web application security.

Everyone gets hacked, from large consumer e-commerce sites and portals, such as Yahoo!, to government agencies, such as the National Aeronautics and Space Administration (NASA) and the Central Intelligence Agency (CIA). In the past, the majority of security breaches occurred at the network layer of enterprise systems. Today, however, hackers are manipulating web applications inside the enterprise firewall, enabling them to access and sabotage corporate and customer data. Given even a tiny vulnerability in a company's web application code, an experienced intruder with only a web browser and a little determination can break into most commercial websites.

The problem is much greater than industry watchdogs realize.

Many businesses do not monitor online activities at the web-application level. This lack of security permits attempted attacks to go unnoticed. It puts organizations into a reactive security posture, where nothing gets fixed until after a situation occurs. Reactive security can mean sacrificing sensitive data as a catalyst for policy change.

Figure 1. Most organizations have a reactive security posture for their mission-critical web applications.



Why aren't web environments secure?

As more organizations take advantage of the Internet, they discover that the web is not just a new market or distribution channel but also a new operating environment. In this new environment, conventional security measures are outdated and frequently ineffective.

A new level of security breach is occurring through continuously open Internet ports (port 80 for general web traffic and port 443 for encrypted traffic). Because these ports are open to all incoming Internet traffic from the outside, they are gateways through which hackers access secure files and proprietary corporate and customer data. While you may read about rogue hackers in the news, the more likely threat is in the form of online theft, terrorism and espionage.

In addition to the vulnerabilities inherent in the Internet operating environment, negligence also accounts for some of the risk to company data. According to the SANS Institute, seven management errors lead to computer security vulnerabilities:

1) Assigning untrained people to maintain security and not providing needed training or time for the job
2) Failing to understand the relationship of information security to the business problem—managers understand physical security but do not see the consequences of poor information security
3) Failing to deal with the operational aspects of security: making a few fixes and then not allowing the follow through needed to make the fixes permanent
4) Relying primarily on a firewall and intrusion detection systems (IDS)
5) Failing to realize how much money information and organizational reputations are worth
6) Authorizing reactive, short-term fixes so problems re-emerge rapidly
7) Pretending the problem will go away

Hackers are one step ahead of the enterprise.

While organizations rush to develop security policies and implement basic security capabilities, professional hackers continue to find new ways to attack. Most hackers use "out-of-the-box" security holes to gain escalated privileges or execute commands on a company's server. Simple misconfigurations of off-the-shelf web applications can leave gaping security vulnerabilities in an unsuspecting company's website.

It's not a question of if your site will be attacked but when.

Attacks on web-connected servers have become more common. For example, attackers stole credit card numbers from the Western Union website, and a computer hacker broke into a Walt Disney Company computer, stealing sensitive guest information. There is also resulting brand deterioration, which Ford experienced when its website was defaced. In each of these highly publicized incidents, attackers used security holes in web-based computer applications to access and steal proprietary data and sensitive information or to make changes to a corporate system.

Passwords are not enough.
Passwords are only as secure as the people using them. If you only rely on passwords to protect access to your data, then you are relying entirely on the people creating and using those passwords. Historically, the easiest way to penetrate a company's defenses is with inside knowledge. For example, if a hacker knows that "Bob" works for your company and that "Bob" has a family website with information such as the names and ages of his children, odds are that the hacker can determine Bob's password and be inside your company's defenses very quickly. Even if the integrity of your passwords remains intact, a hidden backup file, an upload file, a form or even an application running on your web server may allow hackers to identify or bypass your password security.

SSL and data encryption are not enough.
The Secure Sockets Layer (SSL) protocol and data encryption may protect your information during transmission, but when this information is used by your systems, it must be in a readable form. If the applications that process data on your site store information for their own purposes, they probably do not store it in an encrypted format. It is easy to retrieve data from many web-based applications. If your site is vulnerable, then so is your data.

Firewalls are not enough.
When medieval architects designed castles, they spent more time focusing on the gates and the moat than on any other feature. They knew that a defensive system is only as strong as its weakest point. To be useful to your employees with legitimate business needs, your IT environment must allow access from the outside. Such openings provide potential vulnerabilities. The challenge is to provide access only to the people who need it. One of the hardest attacks to recognize and defend against is one that uses your own programs and systems against you. This Trojan-horse type of attack manipulates the features of your own software to force it to divulge information. Firewalls do not prevent this from happening.

Standard scanning programs are not enough.
Most scanners run a series of routine checks against your network, searching for known vulnerabilities on standard servers and applications. Most of these products evaluate your security based on a static list of potential vulnerabilities. All they can tell you is that access to your data may be possible. They do not assess the security of your entire web presence by checking website content (HTML pages, scripts, proprietary applications, cookies and other web servers). As a result, they do not tell you what data a hacker can access or what damage the hacker may inflict on your site and your enterprise by entering through a web application.

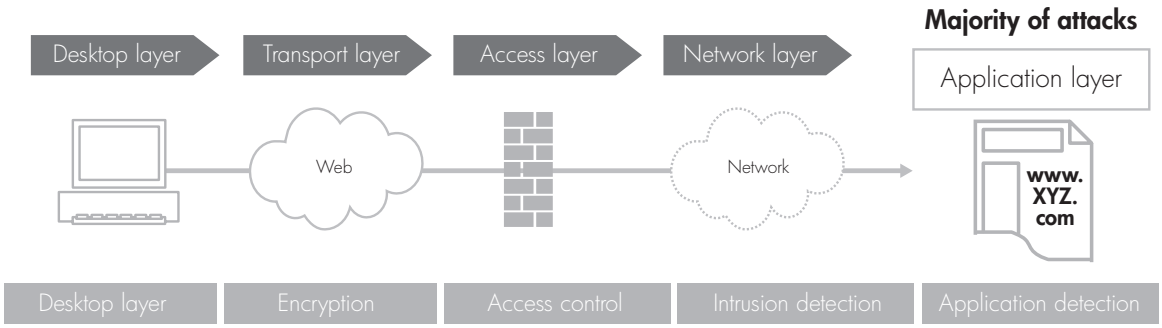
A chain is only as strong as its weakest link.
Even if you have the best internal controls, what about other links in your Internet supply chain: internet service providers (ISPs), application service providers (ASPs), tool vendors and development partners? Can you rely on them to take the same care with your data and information as you do? One example is software licensing agreements. Users who deploy software products are generally required to sign a document stating that any damage that a third-party software product may cause is not the responsibility of the product vendor. In essence, these vendors are saying that security is your problem.

It's in the code.
Programmers typically do not develop web applications with security in mind. Additionally, most companies continue to outsource the majority of their website or web application development, using third-party development resources. Whether these development groups are individuals or consultants, most programmers focus on the "feature and function" side of the development plan and assume that security is embedded into coding practices. However, third-party development resources typically do not have security expertise. They also have specific objectives, such as rapid development schedules, that do not support the security scrutiny required to implement a safe solution.

Manipulating a web application is simple.
Hackers can easily find and change hidden fields that indicate a product price. Using a similar technique, hackers can change the parameters of a Common Gateway Interface (CGI) script to search for a password file instead of a product price. If some components of a web application, such as search functionality, are not integrated and configured correctly, the site may be subject to buffer overflow attacks that can grant hackers access to administrative pages. Today's web application coding practices largely ignore many basic security measures required to keep a company and its data safe from unauthorized access.

A firewall, an intrusion detection system (IDS), cryptography and access control are not enough. Figure 2 shows the progression of a professional hacker through the hacker's value chain. If the hacker appears to be a "normal user," he or she can pass all the regular security checks and become engaged at the application layer. (A visitor to your company's public website appears as a normal user.) Once the hacker reaches the application layer, the attack can begin.

Figure 2. A hacker progresses through an enterprise and into enterprise web applications.



How do you protect your site?

The dynamic nature of the web is most apparent in the area of security. New software and standards for the web are constantly being introduced. Each innovation introduces a potential weakness that hackers can exploit to compromise your network's integrity. In the rush to bring new software products to market, few companies test the security of their products, yet users rely on these products to conduct business every day.

The cost of poor application security can be far greater than most organizations imagine. Not only do you risk your brand and customer data, but common denial of service attacks can prevent you from conducting business.

Even with the best conventional security systems available today, you are likely to be vulnerable to web-based application hacking.

What do you need to do?

Your developers and security professionals must be able to detect holes in both standard and proprietary applications. They can evaluate the severity of the security holes and propose prioritized solutions, protecting existing applications and implementing new software quickly. A typical process involves evaluating all applications on web-connected devices and examining each line of application logic for existing and potential security vulnerabilities.

Unfortunately, most security products cannot adequately examine the applications residing on your web servers, yet these applications often provide back-end access to confidential data. This means you need to be proactive in protecting your critical web applications.

Anatomy of a web application attack

To prevent an application attack, you must first understand how a hacker thinks.

Act 1: The scan
A hacker begins by running a port scan to detect the open HTTP and HTTPS ports for each server and then retrieves the default page from each open port.

Act 2: Information gathering
The hacker then identifies the type of server running on each port and parses each page to find normal links (HTML anchors). This lets the hacker determine the structure of the site and the logic of the application. The hacker then analyzes the found pages and checks for comments and other useful data that may reference files and directories not intended for public use.

Act 3: Testing
The hacker goes through a testing process for each of the application scripts or dynamic functions of the application, looking for development errors in order to gain further access into the application.

Act 4: Planning the attack
After identifying all of the information available through passive (undetectable) means, the hacker selects and deploys attacks. These attacks focus on the information gained from the passive, information-gathering process.

Act 5: Launching the attack
The hacker then attacks each web application identified as vulnerable during the initial review of your site.

The results of the attack can include lost data, content manipulation, theft and loss of customers. The average corporation cannot detect such attacks and may use significant resources trying to diagnose the implications of an attack.

The potential for loss is significant. A hacker can copy sensitive corporate information, such as proprietary customer databases or records, and disseminate that information to competitors or to the general public without your knowledge.

Attack techniques

A hacker can use a variety of techniques to exploit your web application, including the following:

- Parameter manipulation can be as simple as an invalid value passed to the web application so that the application reveals some internal data about itself. It can also be as complex as passing a hidden Structured Query Language (SQL) statement that can access data from your database.
- Forcing a parameter attempts to exploit the programming rather than the application by determining debugging and testing flags. When flags are present, they may be used to enable special, normally hidden modes within the application.
- Cookie tampering manipulates the contents of cookies passed between users and the web application. This tampering can result in an application permitting access to an unauthorized user, or the application may mishandle the contents of the cookie and return restricted data.
- Common file query looks for files that have been inadvertently left accessible by developers, administrators or default application configurations. The result can be the exposure of sensitive information that otherwise should have been removed from the application.

These are just a few examples of the types of attacks that professional hackers may attempt.

Common application attack methodologies

Attacks primarily fall into two types: static and dynamic. Static attacks are commonly known attack methods, while dynamic attacks are harder to detect and protect against because they are launched from deep within the application logic. This section provides a more complete list of attacks and how they operate along with suggested remediation techniques.

The attacks in this section are known by the hacker community and result from poor application coding practices, sub-par administration processes or application misconfigurations.

Static vulnerability attacks
Static vulnerability attacks include:

- Known exploits
- Directory enumeration
- Web server testing

Known exploits
Hackers post the latest discovered web application attacks on a variety of forums and "underground" websites that contain information for the black-hat hacker community. The number of postings is in the thousands with many more discovered and posted on a daily basis. A hacker can use any of the known exploits to attack your web application. For example, a hacker can use exploits such as:

- Remote database services (RDS) exploit
- Code red worm exploit
- Nimda virus

Remediation: Apply all general application patches if available. If not available, you need to conduct a secure code audit and review to detect vulnerabilities in your application code.

Directory enumeration
A hacker attempts to map your website hierarchy and directory structure by looking for common directory names that are hidden from public view but are accessible. These directories can contain administrative pages or sensitive information that can help the attacker further access into the application.

Remediation: Keep the web root of your web servers clean by removing hidden directories. Check that the servers only contain public content. If you need hidden directories, make sure that they are protected by proper authentication mechanisms.

Web server testing
Many web servers have vulnerabilities that allow attackers to submit malformed requests. These can result in unauthorized access to the system.

Remediation: If the servers have patches, make sure you have installed the latest updates.

Dynamic vulnerability attacks
The majority of security products stop being effective when faced with dynamic vulnerability attacks because they do not include these types of security checks. They only detect known vulnerabilities in a certain location or within a fixed path (for example, the /cgi-bin directory).

A dynamic vulnerability attack uncovers vulnerabilities even if they are deep within your web structure. For example, standard fixed vulnerability products search for vulnerabilities within a fixed path (such as /cgi-bin/formmail.pl). But what if formmail.pl is located in /scripts/mail/formmail.pl? A hacker can find formmail.pl regardless of its location in your web structure. Attackers know that it is an exploitable script even though it is not in its default location. You need to review the directory tree to confirm that your web application does not expose vulnerable files.

Dynamic vulnerability attacks include the following:

- Link traversal
- Path truncation
- Session hijacking
- Hidden web paths
- Java™ applet reverse engineering
- Backup checking
- Extension checking
- Parameter passing
- Common file checks
- Cross-site scripting (XSS)
- SQL injection

Link traversal
A hacker "crawls" a web application to define its structure and logic flow. This is an information-gathering attack and is usually the initial step in a series of attacks. It enables a hacker to identify URLs that may no longer be in production but are still referenced in commented-out sections of your web application.

Remediation: Analyze your link structure and confirm that unnecessary links are removed from public access.

Path truncation
In this information-gathering attack, the hacker requests only the directories found in a site but not specific files. If your web server does not have a default page located within a directory or specified in the web server configuration, the hacker sees the contents of the directory. This provides the attacker a significant amount of information about the application and its structure.

For example, if your site links to /customers/id/993/details.html, the hacker can begin truncating the path, looking for vulnerabilities as follows:

First truncation: /customers/id/993/

Second truncation: /customers/id/

Third truncation: /customers/

This frequently uncovers vulnerabilities that neither development staff nor network administrators know exist.

Remediation: Confirm that a default file is located within each directory, and disable directory listings in the web server configuration files.

Session hijacking
The hacker can "hijack" another user's session by intercepting or predicting cookies that the site sends. This lets the hacker impersonate an authenticated user and review all information that the authenticated user can review.

Remediation: This vulnerability is usually a design issue within the application, caused by the application generating predictable session IDs or cookies. Review application code, and prohibit predictable session IDs and cookies.

Hidden web paths
The hacker finds hidden paths or references in the source code or comments within a web application. This information can provide access to restricted areas of your web application. For example:

<!— my old path /webroot/old/bleh.asp —>

Remediation: Keep HTML comments on a beta server and remove them before migrating the application into production. Confirm that HTML source code is free of references and comments that relate to the design of the web application code.

Java applet reverse engineering
Java applets are commonly used for client-side log-ons. Vulnerability exists because a Java applet can be decompiled easily. This lets the hacker find paths or links and use that information for obtaining unauthorized information from your website.

Remediation: Do not use Java applets as the sole means of authentication or access control. Also, use obfuscation to make it difficult for an attacker to decompile the applet.

Backup and extension checking
The hacker works from a list of commonly used backup folder names and file extensions. If any of these folders exists on the site, the hacker searches for a mirror of the original site within that folder. If the hacker finds a mirrored site, the hacker parses through the site, looking for valid information that can lead to a security breach.

For example, directories such as /backup and /oldsite may contain old revisions of your company's site. Though these sites are no longer used, they may contain pages or scripts that can lead to information that compromises your site.

Remediation: Delete all backup files from production servers, and disable mechanisms that automatically create backup files on a production server. If you need to store backup files on your production servers, move the files out of the web tree to prevent unauthorized access.

Parameter passing
Many vulnerabilities exist because application developers and site designers fail to consider how hostile users can add data to web page form fields. The hacker evaluates the parameters used by scripts and enters invalid or user-specified values.

For example, a hacker can use the following script:

/contact.asp?email=bleh@bleh.com&subject=Send+me+stuff,

The hacker may attempt many variations, such as:

/contact.asp?email='&subject=

/contact.asp?email=|ls&subject=

/contact.asp?email=/../../../etc/passwd&subject=/../../../etc/passwd

This technique also checks for buffer overflows in the application.

Remediation: Validate all user input from your servers. Do not rely on client JavaScript validation.

Common file checks
Many sites have common file names or common directory names within site directories. For example, many administrators use WS-FTP to upload updated files to their web sites. However, WS-FTP leaves a file called WS_FTP.LOG in the directory to which the files are uploaded.

These common files provide information that attackers can use to compromise your website. A hacker searches through every directory for these common files and uses this information to attack your site.

Remediation: Remove these files from the production server web tree.

Cross-site scripting (XSS)
A hacker forces a web server to serve JavaScript that was not originated by the website's authors. This malicious JavaScript can steal a user's cookies and compromise the user's computer.

Remediation: Deny any HTML input to your web application.

SQL injection
SQL injection exploits web applications by using client-supplied data in SQL queries without first stripping illegal characters. The hacker places SQL commands into web page forms or parameters. The hacker may run SQL commands on your database, compromising your database server. Although it is simple to protect against SQL injection, a large number of production systems connected to the Internet are vulnerable to this type of attack.

Remediation: Parse and filter all input.

About HP WebInspect

HP WebInspect software helps you safeguard your entire network with intuitive, intelligent and accurate processes that dynamically scan standard and proprietary web applications for known and unidentified application vulnerabilities. HP WebInspect provides a new level of protection for your critical business information. With HP WebInspect, you can find and correct vulnerabilities at their source before attackers can exploit them.

Whether you are an application developer, security auditor, QA professional or security consultant, HP WebInspect provides the capabilities you need for verifying the security of your web applications. HP WebInspect addresses the complexity of Web 2.0 and new web technologies, such as Ajax, and identifies vulnerabilities that are undetectable by traditional scanners. HP WebInspect tackles today's most complex web application technologies with breakthrough testing innovations, including simultaneous crawl and audit (SCA) and concurrent application scanning, resulting in faster and more accurate automated web application security testing. HP WebInspect lets you perform security assessments for any web application, including these industry-leading application platforms:

- Macromedia ColdFusion
- Lotus Domino
- Oracle® Application Server
- Macromedia JRun
- BEA WebLogic
- Jakarta Tomcat

Conclusion

One white paper cannot describe all of the remedies for corporate security issues. This paper introduces the critical issues that exist when your organization engages the power of the Internet. You must keep current on all aspects of security if you intend to stay ahead of professional hackers and protect your critical data assets.

The sooner you can detect, assess and remediate your web application security vulnerabilities, the less likely you become a victim of online fraud. A proactive approach helps you confirm that your intellectual property is not at risk. In summary, you must know what the hackers know.

Business case for application security

Whether a security breach is made public or confined internally, the fact that a hacker can access your sensitive data is a huge concern to your company, your shareholders and most importantly, your customers. Companies that are vigilant and proactive in their approach to application security are better protected. In the long run, these companies enjoy a higher return on investment for their e-business ventures.

To learn more, visit www.hp.com/go/software

4AA1-5384ENW, October 2007